# Import ไลบรารี csv

import csv

**import csv**: นำเข้าไลบรารี csv เพื่อใช้ในการอ่านและเขียนไฟล์ CSV

# ฟังก์ชันสำหรับเพิ่มห้อง

def add\_room(rooms, room\_number, room\_type, price):

    for room in rooms:

        if room['room\_number'] == room\_number:

            return False

    rooms.append({'room\_number': room\_number, 'room\_type': room\_type, 'price': price})

    return True

ฟังก์ชัน **`add\_room(rooms, room\_number, room\_type, price)`** ใช้สำหรับเพิ่มข้อมูลห้องพักใหม่ลงในรายการห้องพัก (`rooms`) ดังนี้:

1. พารามิเตอร์ **`rooms`:** รายการห้องพักที่ต้องการเพิ่มข้อมูลห้องใหม่ลงไป.

2. พารามิเตอร์ **`room\_number`:** หมายเลขห้องของห้องพักใหม่ที่ต้องการเพิ่ม.

3. พารามิเตอร์ **`room\_type`:** ประเภทห้องของห้องพักใหม่.

4. พารามิเตอร์ **`price`:** ราคาของห้องพักใหม่.

**การทำงานของฟังก์ชัน:**

1. ใช้ลูป **`for`** เพื่อวนลูปผ่านทุกห้องในรายการ **`rooms`.**

2. ที่แต่ละรอบของลูป เช็คว่าหมายเลขห้องของห้องในรอบนั้น **(`room['room\_number']`)** เท่ากับหมายเลขห้องที่ต้องการเพิ่ม (`room\_number`).

3. ถ้าหมายเลขห้องที่เราต้องการเพิ่มตรงกับหมายเลขห้องในรอบนั้น ให้ทำการคืนค่า **`False`** เพื่อแสดงว่าไม่สามารถเพิ่มห้องได้เนื่องจากมีหมายเลขห้องซ้ำ.

4. หากหมายเลขห้องที่เราต้องการเพิ่มไม่ซ้ำกับหมายเลขห้องในรายการ ให้ทำการเพิ่มข้อมูลห้องพักใหม่ลงในรายการ **`rooms`** โดยเพิ่มเป็นพจน์คีย์-แอน-ค่า (key-value pairs) ซึ่งประกอบด้วยหมายเลขห้อง (`room\_number`), ประเภทห้อง (`room\_type`), และราคา (`price`).

5. เมื่อเพิ่มข้อมูลห้องพักใหม่สำเร็จ ใช้ **`return True`** เพื่อแสดงว่าการเพิ่มห้องสำเร็จ.

ฟังก์ชัน **`add\_room`** นี้จะทำการเพิ่มข้อมูลห้องพักใหม่ลงในรายการห้องพัก **(`rooms`)** และจะคืนค่า **`True`** หากการเพิ่มห้องสำเร็จ หรือคืนค่า **`False`** หากห้องที่ต้องการเพิ่มมีหมายเลขซ้ำกับห้องที่มีอยู่ในรายการแล้ว.

# ฟังก์ชันสำหรับจองห้อง

def book\_room(rooms, room\_number):

    for room in rooms:

        if room['room\_number'] == room\_number:

            if not room.get('reserved', False):

                room['reserved'] = True

                return True

            else:

                return False

    return False

ฟังก์ชัน **`book\_room(rooms, room\_number)`** ใช้สำหรับการจองห้องพักที่มีหมายเลขห้องตรงกับ **`room\_number`** ดังนี้:

1. พารามิเตอร์ **`rooms`**: รายการห้องพักที่ต้องการค้นหาและทำการจอง.

2. พารามิเตอร์ **`room\_number`**: หมายเลขห้องที่ต้องการจอง.

**การทำงานของฟังก์ชัน:**

1. ใช้ลูป **`for`** เพื่อวนลูปผ่านทุกห้องในรายการ **`rooms`.**

2. ที่แต่ละรอบของลูป เช็คว่าหมายเลขห้องของห้องในรอบนั้น **(`room['room\_number']`)** เท่ากับหมายเลขห้องที่ต้องการจอง **(`room\_number`).**

3. ถ้าหมายเลขห้องที่เราต้องการจองตรงกับหมายเลขห้องในรอบนั้น ใช้ **`room.get('reserved', False)`** เพื่อตรวจสอบว่าห้องนั้นมีสถานะการจอง **(`reserved`)** เป็น **`True`** หรือไม่ หากไม่มีการจอง ให้กำหนดสถานะการจองเป็น **`True` (`room['reserved'] = True`)** เพื่อทำการจอง.

4. ถ้าห้องที่เราต้องการจองไม่ว่าง (มีสถานะการจองเป็น `True`) ให้ทำการคืนค่า **`False`** เพื่อแสดงว่าไม่สามารถจองห้องได้.

5. ถ้าหมายเลขห้องที่เราต้องการจองไม่ตรงกับหมายเลขห้องในรอบนั้น ไม่ต้องทำอะไรและเราจะเดาว่าห้องที่ต้องการจองไม่มีอยู่ในรายการ **`rooms`** จึงทำการคืนค่า **`False`.**

ฟังก์ชัน **`book\_room`** นี้จะทำการค้นหาและจองห้องพักที่มีหมายเลขห้องตรงกับค่า **`room\_number`** หากห้องว่างและสามารถจองได้ จะคืนค่า **`True`** และหากห้องไม่ว่างหรือไม่สามารถจองได้ จะคืนค่า **`False`** ให้แสดงว่าไม่สามารถจองห้องได้.

# ฟังก์ชันสำหรับยกเลิกการจองห้อง

def cancel\_booking(rooms, room\_number):

    for room in rooms:

        if room['room\_number'] == room\_number and room.get('reserved', False):

            room['reserved'] = False

            return True

    return False

ฟังก์ชัน **`cancel\_booking(rooms, room\_number)`** ใช้สำหรับยกเลิกการจองห้องพักที่มีหมายเลขห้องตรงกับ **`room\_number`** โดยเฉพาะห้องที่มีสถานะการจอง **(`reserved`)** เป็น **`True`** ดังนี้:

1. พารามิเตอร์ **`rooms`**: รายการห้องพักที่ต้องการค้นหาและทำการยกเลิกการจอง.

2. พารามิเตอร์ **`room\_number`**: หมายเลขห้องที่ต้องการยกเลิกการจอง.

**การทำงานของฟังก์ชัน:**

1. ใช้ลูป **`for`** เพื่อวนลูปผ่านทุกห้องในรายการ **`rooms`.**

2. ที่แต่ละรอบของลูป เช็คว่าหมายเลขห้องของห้องในรอบนั้น **(`room['room\_number']`)** เท่ากับหมายเลขห้องที่ต้องการยกเลิกการจอง **(`room\_number`)** และเช็คว่าห้องนั้นมีสถานะการจอง **(`reserved`)** เป็น **`True`** หรือไม่ **(`room.get('reserved', False)`).**

3. ถ้าหมายเลขห้องและสถานะการจองของห้องที่เราต้องการยกเลิกตรงกับหมายเลขห้องและสถานะที่กำหนด ให้เปลี่ยนสถานะการจองเป็น **`False` (`room['reserved'] = False`)** เพื่อยกเลิกการจอง.

4. เมื่อทำการยกเลิกการจองสำเร็จ ใช้ **`return True`** เพื่อแสดงว่าการยกเลิกการจองสำเร็จ.

5. ถ้าหมายเลขห้องและสถานะการจองของห้องที่เราต้องการยกเลิกไม่ตรงกับหมายเลขห้องและสถานะที่กำหนด ไม่ต้องทำอะไรและเราจะเดาว่าห้องนั้นไม่มีการจอง จึงทำการคืนค่า **`False`.**

ฟังก์ชัน **`cancel\_booking`** นี้จะทำการค้นหาและยกเลิกการจองของห้องพักที่มีหมายเลขห้องตรงกับค่า **`room\_number`** และมีสถานะการจองเป็น **`True`** และจะคืนค่า **`True`** หากการยกเลิกการจองสำเร็จ และ **`False`** หากห้องที่ต้องการยกเลิกไม่มีการจองหรือสถานะการจองไม่ถูกต้อง.

# ฟังก์ชันสำหรับแก้ไขข้อมูลห้อง

def edit\_room(rooms, room\_number, new\_room\_type, new\_price):

    for room in rooms:

        if room['room\_number'] == room\_number:

            room['room\_type'] = new\_room\_type

            room['price'] = new\_price

            return True

    return False

ฟังก์ชัน **`edit\_room(rooms, room\_number, new\_room\_type, new\_price)`** ใช้สำหรับแก้ไขข้อมูลของห้องพักที่มีหมายเลขห้องตรงกับ **`room\_number`** ด้วยข้อมูลใหม่ที่กำหนด ซึ่งประกอบด้วยประเภทห้องใหม่ **(`new\_room\_type`)** และราคาใหม่ **(`new\_price`)** ดังนี้:

1. พารามิเตอร์ **`rooms`**: รายการห้องพักที่ต้องการค้นหาและแก้ไขข้อมูล.

2. พารามิเตอร์ **`room\_number`**: หมายเลขห้องที่ต้องการแก้ไขข้อมูล.

3. พารามิเตอร์ **`new\_room\_type`**: ประเภทห้องใหม่ที่ต้องการกำหนด.

4. พารามิเตอร์ **`new\_price`**: ราคาใหม่ที่ต้องการกำหนด.

**การทำงานของฟังก์ชัน:**

1. ใช้ลูป **`for`** เพื่อวนลูปผ่านทุกห้องในรายการ **`rooms`.**

2. ที่แต่ละรอบของลูป เช็คว่าหมายเลขห้องของห้องในรอบนั้น **(room['room\_number'])** เท่ากับหมายเลขห้องที่ต้องการแก้ไข **(`room\_number`).**

3. ถ้าหมายเลขห้องที่เราต้องการแก้ไขตรงกับหมายเลขห้องในรอบนั้น ใช้การกำหนดค่าใหม่ให้กับประเภทห้อง **(`room['room\_type'] = new\_room\_type`)** และราคา **(`room['price'] = new\_price`)** โดยแก้ไขข้อมูลประเภทห้องและราคาตามที่กำหนด.

4. เมื่อแก้ไขข้อมูลเรียบร้อยแล้ว ใช้ **`return True`** เพื่อแสดงว่าการแก้ไขสำเร็จ.

5. ถ้าหมายเลขห้องที่เราต้องการแก้ไขไม่ตรงกับหมายเลขห้องในรอบนั้น ไม่ต้องทำอะไรและเราจะเดาว่าห้องที่ต้องการแก้ไขไม่อยู่ในรายการ **`rooms`** จึงทำการคืนค่า **`False`.**

ฟังก์ชัน **`edit\_room`** นี้จะทำการค้นหาและแก้ไขข้อมูลห้องพักที่มีหมายเลขห้องตรงกับค่า **`room\_number`** ด้วยข้อมูลใหม่ที่กำหนด และจะคืนค่า **`True`** หากการแก้ไขสำเร็จและ **`False`** หากห้องที่ต้องการแก้ไขไม่อยู่ในรายการ.

# ฟังก์ชันสำหรับลบห้อง

def delete\_room(rooms, room\_number):

    for room in rooms:

        if room['room\_number'] == room\_number:

            rooms.remove(room)

            return True

    return False

ฟังก์ชัน **`delete\_room(rooms, room\_number)`** ใช้สำหรับลบข้อมูลของห้องพักที่มีหมายเลขห้องตรงกับ **`room\_number`** ออกจากรายการ **`rooms`** ดังนี้:

1. พารามิเตอร์ **`rooms`**: รายการห้องพักที่ต้องการค้นหาและลบห้อง.

2. พารามิเตอร์ **`room\_number`**: หมายเลขห้องที่ต้องการลบ.

**การทำงานของฟังก์ชัน:**

1. ใช้ลูป **`for`** เพื่อวนลูปผ่านทุกห้องในรายการ **`rooms`.**

2. ที่แต่ละรอบของลูป เช็คว่าหมายเลขห้องของห้องในรอบนั้น **(room['room\_number'])** เท่ากับหมายเลขห้องที่ต้องการลบ **(`room\_number`).**

3. ถ้าหมายเลขห้องที่เราต้องการลบตรงกับหมายเลขห้องในรอบนั้น ใช้ **`rooms.remove(room)`** เพื่อลบห้องนั้นออกจากรายการ **`rooms`** และทำการคืนค่า **`True`** เพื่อบอกว่าการลบสำเร็จ.

4. ถ้าหมายเลขห้องที่เราต้องการลบไม่ตรงกับหมายเลขห้องในรอบนั้น ไม่ต้องทำอะไรและเราจะเดาว่าห้องที่ต้องการลบไม่มีอยู่ในรายการ **`rooms`** จึงทำการคืนค่า **`False`.**

ฟังก์ชัน **`delete\_room`** นี้จะทำการค้นหาและลบห้องพักที่มีหมายเลขห้องตรงกับค่า **`room\_number`** ออกจากรายการ **`rooms`** และจะคืนค่า **`True`** หากการลบสำเร็จและ **`False`** หากห้องที่ต้องการลบไม่มีในรายการ.

# ฟังก์ชันสำหรับแสดงข้อมูลห้อง

def display\_rooms(rooms, show\_option):

    if show\_option == "1":

        rooms\_to\_display = rooms

    elif show\_option == "2":

        rooms\_to\_display = [room for room in rooms if not room.get('reserved', False)]

    elif show\_option == "3":

        rooms\_to\_display = [room for room in rooms if room.get('reserved', True)]

    elif show\_option == "4":

        search\_room\_number = int(input("กรุณาป้อนเลขห้องที่ต้องการค้นหา: "))

        rooms\_to\_display = [room for room in rooms if room['room\_number'] == search\_room\_number]

    if not rooms\_to\_display:

        print("ไม่มีห้องที่ต้องการแสดงผล")

        return

    sorted\_rooms = selection\_sort(rooms\_to\_display)

    for room in sorted\_rooms:

        print(f"ห้องเลขที่: {room['room\_number']}, ประเภทห้อง: {room['room\_type']}, ราคา: {room['price']}, สถานะ: {'จองแล้ว' if room.get('reserved', False) else 'ว่าง'}")

**ฟังก์ชัน display\_rooms(rooms, show\_option):**

- ใช้สำหรับแสดงข้อมูลห้องตามตัวเลือกที่กำหนด (แสดงห้องทั้งหมด, ห้องที่ว่าง, ห้องที่มีการจองแล้ว)

- หากไม่มีห้องที่ต้องการแสดงผล จะแสดงข้อความ **"ไม่มีห้องที่ต้องการแสดงผล"**

1. พารามิเตอร์ **`rooms`**: รายการของห้องพักทั้งหมดที่มีในโรงแรม.

2. พารามิเตอร์ **`show\_option`**: ตัวเลือกการแสดงผลที่ผู้ใช้เลือก เป็นสตริงที่เป็นตัวเลข 1, 2, หรือ 3.

**การทำงานของฟังก์ชัน:**

1. เช็คเงื่อนไขตามค่า **`show\_option`**:

- ถ้า **`show\_option`** เป็น **"1"**: แสดงทุกห้องที่มีในรายการ โดยกำหนดให้ **`rooms\_to\_display`** เป็นรายการ **`rooms`** ทั้งหมด

- ถ้า **`show\_option`** เป็น **"2"**: แสดงเฉพาะห้องที่ว่าง โดยใช้ List Comprehension ในการกรองห้องที่ไม่มีการจอง แล้วเก็บไว้ใน **`rooms\_to\_display`**

- ถ้า **`show\_option`** เป็น **"3"**: แสดงเฉพาะห้องที่มีการจองแล้ว โดยใช้ List Comprehension ในการกรองห้องที่มีการจอง แล้วเก็บไว้ใน **`rooms\_to\_display`**

- ถ้า **`show\_option`** เป็น **"4"**: จะให้ผู้ใช้งานกรอกเลขห้องที่ต้องการแสดงผล แล้วใช้ List Comprehension กรองห้องด้วยเลขห้องที่ผู้ใช้งานกรอกเข้าไป ถ้าเจอก็จะเก็บข้อมูลห้องนั้นไว้ใน **‘rooms\_to\_display’**

2. เช็คว่า **`rooms\_to\_display`** มีห้องหรือไม่ ถ้าไม่มีจะแสดงข้อความ **"ไม่มีห้องที่ต้องการแสดงผล"** และจะจบการทำงานของฟังก์ชัน

3. นำ **`rooms\_to\_display`** ที่ถูกกรองและกำหนดตามตัวเลือกการแสดงผลมาทำการเรียงลำดับด้วยฟังก์ชัน **`selection\_sort`** และเก็บผลลัพธ์ไว้ใน **`sorted\_rooms`**

4. ใช้ลูป **`for`** เพื่อวนเข้าถึงทุกห้องใน **`sorted\_rooms`** และแสดงข้อมูลของแต่ละห้อง โดยรวมถึงหมายเลขห้อง (room\_number), ประเภทห้อง (room\_type), ราคา (price), และสถานะ (reserved หรือ ว่าง) โดยใช้เงื่อนไข **`if`** เพื่อตรวจสอบสถานะของการจอง ถ้าห้องถูกจองแล้วจะแสดง **"จองแล้ว"** ถ้าห้องว่างจะแสดง **"ว่าง"**

ฟังก์ชัน **display\_rooms** นี้เป็นตัวกลางที่จัดการการแสดงข้อมูลห้องพักตามตัวเลือกที่ผู้ใช้เลือก และเรียกใช้ฟังก์ชันอื่น ๆ เช่น **selection\_sort** เพื่อให้ผู้ใช้เห็นข้อมูลห้องพักที่เรียงลำดับแล้วและได้แสดงสถานการณ์จองห้องด้วย

# ฟังก์ชันสำหรับค้นหาห้องตามประเภทห้อง

def search\_by\_type(rooms, room\_type):

    found\_rooms = [room for room in rooms if room['room\_type'] == room\_type]

    if not found\_rooms:

        print("ไม่พบห้องที่ต้องการค้นหา")

        return

    print("รูปแบบการแสดงผล")

    print("1. แสดงห้องทั้งหมด")

    print("2. แสดงห้องที่ยังไม่มีคนจอง")

    print("3. แสดงห้องที่มีคนจองแล้ว")

    show\_option = input("กรุณาเลือกรูปแบบการแสดงผล : ")

    if show\_option in ["1","2","3"]:

        display\_rooms(found\_rooms,show\_option)

    else:

        print("ท่านเลือกไม่ถูกต้องกรุณาเลือกแค่ 1,2,3")

**ฟังก์ชัน search\_by\_type(rooms, room\_type)** ใช้สำหรับค้นหาห้องพักตามประเภทที่กำหนด และแสดงผลตามตัวเลือกการแสดงผลที่ผู้ใช้เลือก ดังนี้:

1. พารามิเตอร์ **`rooms`**: รายการของห้องพักทั้งหมดในโรงแรม.

2. พารามิเตอร์ **`room\_type`**: ประเภทห้องที่ผู้ใช้ต้องการค้นหา.

**การทำงานของฟังก์ชัน:**

1. ใช้ List Comprehension เพื่อกรองห้องพักในรายการ **`rooms`** ที่มีประเภทห้องเหมือนกับ **`room\_type`** แล้วเก็บไว้ในรายการ **`found\_rooms`.**

2. เช็คว่า **`found\_rooms`** มีห้องหรือไม่ ถ้าไม่มีจะแสดงข้อความ **"ไม่พบห้องที่ต้องการค้นหา"** และจะจบการทำงานของฟังก์ชัน.

3. แสดงเมนูการแสดงผลที่ผู้ใช้สามารถเลือกได้ โดยแสดงตัวเลือก **"แสดงห้องทั้งหมด"**, **"แสดงห้องที่ยังไม่มีคนจอง"**, และ **"แสดงห้องที่มีคนจองแล้ว".**

4. รับค่าตัวเลือกการแสดงผลจากผู้ใช้ผ่าน **`input`.**

5. ตรวจสอบว่าตัวเลือกการแสดงผลที่ผู้ใช้เลือกอยู่ในช่วง 1-3 หรือไม่ ถ้าใช่ให้เรียกใช้ฟังก์ชัน **`display\_rooms(found\_rooms, show\_option)`** เพื่อแสดงผลห้องพักที่ถูกค้นพบตามประเภทและตัวเลือกการแสดงผลที่เลือก.

6. ถ้าตัวเลือกการแสดงผลไม่ถูกต้อง (ไม่อยู่ในช่วง 1-3) จะแสดงข้อความ **"ท่านเลือกไม่ถูกต้องกรุณาเลือกแค่ 1,2,3".**

# ฟังก์ชันสำหรับการจัดเรียงห้องด้วย Selection Sort

def selection\_sort(rooms):

    n = len(rooms)

    for i in range(n - 1):

        min\_idx = i

        for j in range(i + 1, n):

            if rooms[j]['room\_number'] < rooms[min\_idx]['room\_number']:

                min\_idx = j

        rooms[i], rooms[min\_idx] = rooms[min\_idx], rooms[i]

    return rooms

ฟังก์ชัน **selection\_sort(rooms)** ใช้วิธีการเรียงลำดับแบบ Selection Sort เพื่อจัดเรียงลำดับห้องพักในรายการ **`rooms`** ตามหมายเลขห้องพัก (room\_number) จากน้อยไปหามาก ดังนี้:

1. พารามิเตอร์ **`rooms`:** รายการห้องพักที่ต้องการจัดเรียงลำดับ.

**การทำงานของฟังก์ชัน:**

1. ใช้ตัวแปร **`n`** เก็บจำนวนห้องในรายการ **`rooms`.**

2. ใช้ลูป **`for`** เพื่อทำการวนลูปเป็นจำนวน **`n - 1`** รอบ (เพื่อให้ห้องสุดท้ายไม่ต้องเปรียบเทียบเพราะเมื่อห้องทั้งหมดถูกจัดเรียงแล้ว ห้องสุดท้ายจะเป็นห้องที่มีหมายเลขห้องมากที่สุด).

3. กำหนดค่าตัวแปร **`min\_idx`** เพื่อเก็บดัชนีของห้องที่มีหมายเลขห้องน้อยที่สุดในช่วงที่ยังไม่ถูกจัดเรียง.

4. ใช้ลูป **`for`** รอบที่สอง (เริ่มจาก **`i + 1`** ไปจนถึง **`n`**) เพื่อวนลูปเพื่อหาห้องที่มีหมายเลขห้องน้อยที่สุดในช่วงที่ยังไม่ถูกจัดเรียง และเก็บดัชนีของห้องเหล่านี้ใน **`min\_idx`.**

5. เมื่อหาห้องที่มีหมายเลขห้องน้อยที่สุดในช่วงที่ยังไม่ถูกจัดเรียงเสร็จแล้ว ทำการสลับห้องที่อยู่ในตำแหน่ง **`i`** และห้องที่มีหมายเลขห้องน้อยที่สุด (ที่อยู่ในตำแหน่ง **`min\_idx`**) โดยใช้การกำหนดค่าพร้อมกัน (tuple assignment).

6. เมื่อลูปใน **`for`** รอบที่สามเสร็จสิ้น ห้องที่มีหมายเลขห้องน้อยที่สุดในช่วงที่เราต้องการจัดเรียงจะอยู่ในตำแหน่ง **`i`.**

7. ทำการเรียกคืนรายการ **`rooms`** ที่ถูกจัดเรียงลำดับแล้ว.

การทำงานของฟังก์ชันนี้จะทำให้รายการ **`rooms`** ถูกจัดเรียงลำดับตามหมายเลขห้องพักจากน้อยไปหามาก ซึ่งเป็นการเตรียมข้อมูลสำหรับการแสดงผลห้องพักให้มีลำดับที่เรียบร้อย.

# ฟังก์ชันสำหรับบันทึกข้อมูลห้องลงในไฟล์ CSV

def save\_to\_csv(rooms, filename):

    with open(filename, mode='w', newline='') as file:

        writer = csv.writer(file)

        writer.writerow(["ห้องเลขที่", "ประเภทห้อง", "ราคา", "สถานะ"])

        for room in rooms:

            writer.writerow([room['room\_number'], room['room\_type'], room['price'], 'จองแล้ว' if room.get('reserved', False) else 'ว่าง'])

ฟังก์ชัน **`save\_to\_csv(rooms, filename)`** ใช้สำหรับบันทึกข้อมูลห้องพักจากรายการ **`rooms`** ลงในไฟล์ CSV โดยให้ผลลัพธ์ที่บันทึกมีรายละเอียดดังนี้:

1. พารามิเตอร์ **`rooms`**: รายการของห้องพักที่ต้องการบันทึกลงในไฟล์ CSV.

2. พารามิเตอร์ **`filename`**: ชื่อไฟล์ CSV ที่ต้องการบันทึกข้อมูลลงไป.

การทำงานของฟังก์ชัน:

1. ใช้คำสั่ง **`with open()`** เพื่อเปิดไฟล์ **`filename`** สำหรับเขียน **(mode='w')** และกำหนด **`newline=''`** เพื่อจัดการกับการเขียนข้อมูลแบบ CSV ให้ถูกต้อง.

2. สร้างตัวแปร **`writer`** ที่เป็นตัวเขียนข้อมูล CSV โดยใช้ **`csv.writer(file)`** เพื่อเตรียมใช้ในการเขียนข้อมูลลงในไฟล์.

3. เริ่มต้นการเขียนข้อมูล CSV โดยเรียงลำดับแต่ละคอลัมน์ด้วย **`writerow(["ห้องเลขที่", "ประเภทห้อง", "ราคา", "สถานะ"])`**.

4. ใช้ลูป **`for`** เพื่อวนลูปผ่านทุกห้องในรายการ **`rooms`** เพื่อเขียนข้อมูลของแต่ละห้องลงในไฟล์ CSV:

- ใช้ **`writer.writerow()`** เพื่อเขียนข้อมูลของแต่ละห้อง โดยแปลงค่าหมายเลขห้อง (room\_number), ประเภทห้อง (room\_type), ราคา (price), และสถานะการจอง (reserved) เป็นแถวของข้อมูล CSV.

- สำหรับสถานะการจอง เราใช้ **`room.get('reserved', False)`** เพื่อเช็คว่าห้องมีการจองหรือไม่ ถ้ามีจะให้ค่าเป็น **'จองแล้ว' ถ้าไม่มีจะให้ค่าเป็น 'ว่าง'.**

5. เมื่อลูป **`for`** เสร็จสิ้น รายการ **`rooms`** จะถูกบันทึกลงในไฟล์ CSV และไฟล์จะถูกปิดอัตโนมัติเนื่องจากใช้คำสั่ง **`with open()`.**

ฟังก์ชัน **`save\_to\_csv`** นี้จะทำการเขียนข้อมูลห้องพักจากรายการ **`rooms`** ลงในไฟล์ CSV โดยมีรายละเอียดเป็นคอลัมน์ห้องเลขที่, ประเภทห้อง, ราคา, และสถานะการจอง.

# ฟังก์ชันสำหรับโหลดข้อมูลห้องจากไฟล์ CSV

def load\_from\_csv(filename):

    rooms = []

    try:

        with open(filename, mode='r') as file:

            reader = csv.reader(file)

            header = next(reader, None)

            if header:

                for row in reader:

                    room\_number, room\_type, price, status = row

                    rooms.append({'room\_number': int(room\_number), 'room\_type': room\_type, 'price': float(price), 'reserved': status == 'จองแล้ว'})

    except FileNotFoundError:

        pass

    return rooms

ฟังก์ชัน **`load\_from\_csv(filename)`** ใช้สำหรับโหลดข้อมูลห้องพักจากไฟล์ CSV เข้าสู่รายการ **`rooms`** โดยใช้รูปแบบที่ได้กำหนดไว้ในฟังก์ชัน **`save\_to\_csv`** มาเพื่ออ่านและแปลงข้อมูลกลับเป็นรายการห้องพักดั้งเดิม ดังนี้:

1. พารามิเตอร์ **`filename`**: ชื่อไฟล์ CSV ที่ต้องการโหลดข้อมูลมา.

การทำงานของฟังก์ชัน:

1. ใช้คำสั่ง **`with open()`** เพื่อเปิดไฟล์ **`filename`** สำหรับอ่าน **(mode='r')** และเรียกใช้ **`csv.reader(file)`** เพื่อเตรียมอ่านข้อมูลจากไฟล์.

2. ใช้คำสั่ง **`next(reader, None)`** เพื่ออ่านแถวแรกของไฟล์ CSV (header) และกำหนดให้เป็น **`header`** โดยไม่ทำอะไรถ้าไม่พบแถวแรก (หัวตาราง).

3. ใช้ลูป **`for`** เพื่อวนลูปผ่านแต่ละแถวของไฟล์ CSV โดยใช้ **`reader`**:

- อ่านค่าหมายเลขห้อง (room\_number), ประเภทห้อง (room\_type), ราคา (price), และสถานะการจอง (status) จากแถว.

- เพิ่มข้อมูลของห้องพักลงในรายการ **`rooms`** โดยแปลงหมายเลขห้องเป็นตัวเลขจำนวนเต็ม (int) และราคาเป็นตัวเลขทศนิยม (float), รวมถึงการแปลงสถานะการจองเป็น **`True`** หรือ **`False`** โดยอิงจากค่า **`status`.**

4. ถ้าพบข้อผิดพลาดเมื่อพยายามเปิดไฟล์ CSV (FileNotFoundError) จะไม่ต้องทำอะไร และฟังก์ชันจะคืนค่า **`rooms`** ที่ว่างเปล่า.

5. ทำการคืนค่ารายการ **`rooms`** ที่มีข้อมูลห้องพักที่ถูกโหลดจากไฟล์ CSV แล้ว.

ฟังก์ชัน **`load\_from\_csv`** นี้ทำหน้าที่อ่านข้อมูลห้องพักจากไฟล์ CSV และแปลงกลับเป็นรายการห้องพักตามรูปแบบที่ถูกบันทึกไว้ในไฟล์ CSV มาเพื่อนำกลับมาใช้งานในโปรแกรม.

**ส่วนของโปรแกรมหลัก Main**

# โปรแกรมหลัก

def main():

    filename = 'hotel\_rooms.csv'

    rooms = load\_from\_csv(filename)

    while True:

        print("\n===== โปรแกรมจัดการข้อมูลการจองห้องพักโรงแรม =====")

        print("1. เพิ่มห้อง")

        print("2. จองห้อง")

        print("3. ยกเลิกการจองห้อง")

        print("4. แก้ไขข้อมูลห้อง")

        print("5. ลบห้อง")

        print("6. แสดงผลห้อง")

        print("7. ค้นหาห้องตามประเภท")

        print("8. บันทึกข้อมูล")

        print("9. ออกจากโปรแกรม")

        choice = input("กรุณาเลือกทำรายการ: ")

        if choice == '1':

            room\_number = int(input("เลขห้อง: "))

            room\_type = input("ประเภทห้อง: ")

            price = float(input("ราคา: "))

            if add\_room(rooms, room\_number, room\_type, price):

                print("เพิ่มห้องสำเร็จ")

            else:

                print("ไม่สามารถเพิ่มห้องได้ (เลขห้องซ้ำ)")

        elif choice == '2':

            room\_number = int(input("เลขห้องที่ต้องการจอง: "))

            if book\_room(rooms, room\_number):

                print("จองห้องสำเร็จ")

            else:

                print("ไม่สามารถจองห้องได้ (ห้องไม่ว่างหรือมีคนจองแล้ว)")

        elif choice == '3':

            room\_number = int(input("เลขห้องที่ต้องการยกเลิกการจอง: "))

            if cancel\_booking(rooms, room\_number):

                print("ยกเลิกการจองสำเร็จ")

            else:

                print("ไม่สามารถยกเลิกการจองได้ (ห้องนี้ยังไม่ถูกจอง)")

        elif choice == '4':

            room\_number = int(input("เลขห้องที่ต้องการแก้ไข: "))

            found\_room = None

            for room in rooms:

                if room['room\_number'] == room\_number:

                    found\_room = room

                    break

            if found\_room:

                new\_room\_type = input("ประเภทห้องใหม่: ")

                new\_price = float(input("ราคาใหม่: "))

                edit\_room(rooms, room\_number, new\_room\_type, new\_price)

                print("แก้ไขข้อมูลห้องสำเร็จ")

            else:

                print("ไม่พบห้องที่ต้องการแก้ไข")

        elif choice == '5':

            room\_number = int(input("เลขห้องที่ต้องการลบ: "))

            if delete\_room(rooms, room\_number):

                print("ลบห้องสำเร็จ")

            else:

                print("ไม่พบห้องที่ต้องการลบ")

        elif choice == '6':

            print("รูปแบบการแสดงผล")

            print("1. แสดงห้องทั้งหมด")

            print("2. แสดงห้องที่ยังไม่มีคนจอง")

            print("3. แสดงห้องที่มีคนจองแล้ว")

            print("4. แสดงผลเฉพาะเลขห้องที่ต้องการ")

            show\_option = input("กรุณาเลือกรูปแบบการแสดงผล : ")

            if show\_option in ["1","2","3","4"]:

                display\_rooms(rooms,show\_option)

            else:

                print("ท่านเลือกไม่ถูกต้องกรุณาเลือกแค่ 1,2,3")

        elif choice == '7':

            room\_type = input("ประเภทห้องที่ต้องการค้นหา: ")

            search\_by\_type(rooms, room\_type)

        elif choice == '8':

            save\_to\_csv(rooms, filename)

            print("บันทึกข้อมูลสำเร็จ")

        elif choice == '9':

            print("ออกจากโปรแกรม")

            break

        else:

            print("กรุณาเลือกทำรายการใหม่")

1. โปรแกรมเริ่มต้นด้วยการโหลดข้อมูลห้องพักจากไฟล์ CSV โดยใช้ฟังก์ชัน **`load\_from\_csv`** เพื่อเตรียมรายการห้องพักในระบบ.

2. โปรแกรมเริ่มวนลูปไว้เพื่อให้ผู้ใช้ป้อนคำสั่งเพื่อทำรายการต่าง ๆ ในระบบ โดยจะแสดงเมนูทางหน้าจอเพื่อให้ผู้ใช้เลือกทำรายการตามที่ต้องการ.

3. โปรแกรมให้ผู้ใช้ป้อนตัวเลือกโดยใช้คำสั่ง **`input`** และจากนั้นดำเนินการตามคำสั่งที่ผู้ใช้เลือก.

4. **หากผู้ใช้เลือก '1' (เพิ่มห้อง),** โปรแกรมจะขอข้อมูลเลขห้อง, ประเภทห้อง, และราคา เพื่อทำการเพิ่มห้องใหม่ในรายการ หากสามารถเพิ่มห้องได้ โปรแกรมจะแสดงข้อความ "เพิ่มห้องสำเร็จ" ถ้าไม่สามารถเพิ่มห้องได้เนื่องจากเลขห้องซ้ำ โปรแกรมจะแสดงข้อความ "ไม่สามารถเพิ่มห้องได้ (เลขห้องซ้ำ)"

5**. หากผู้ใช้เลือก '2' (จองห้อง),** โปรแกรมจะขอเลขห้องที่ต้องการจอง และจากนั้นตรวจสอบว่าสามารถจองห้องได้หรือไม่ หากสามารถจองห้องได้ โปรแกรมจะแสดงข้อความ "จองห้องสำเร็จ" ถ้าไม่สามารถจองห้องได้เนื่องจากห้องไม่ว่างหรือมีคนจองแล้ว โปรแกรมจะแสดงข้อความ "ไม่สามารถจองห้องได้ (ห้องไม่ว่างหรือมีคนจองแล้ว)"

6**. หากผู้ใช้เลือก '3' (ยกเลิกการจองห้อง),** โปรแกรมจะขอเลขห้องที่ต้องการยกเลิกการจอง และจากนั้นตรวจสอบว่าสามารถยกเลิกการจองห้องได้หรือไม่ หากสามารถยกเลิกการจองห้องได้ โปรแกรมจะแสดงข้อความ "ยกเลิกการจองสำเร็จ" ถ้าไม่สามารถยกเลิกการจองได้เนื่องจากห้องนี้ยังไม่ถูกจอง โปรแกรมจะแสดงข้อความ "ไม่สามารถยกเลิกการจองได้ (ห้องนี้ยังไม่ถูกจอง)"

7. **หากผู้ใช้เลือก '4' (แก้ไขข้อมูลห้อง),** โปรแกรมจะขอเลขห้องที่ต้องการแก้ไข และจากนั้นตรวจสอบว่าห้องที่เลือกมีอยู่ในรายการห้องพักหรือไม่ หากพบห้องที่ต้องการแก้ไข โปรแกรมจะขอประเภทห้องใหม่และราคาใหม่ เพื่อทำการแก้ไขข้อมูลห้อง และแสดงข้อความ "แก้ไขข้อมูลห้องสำเร็จ" ถ้าไม่พบห้องที่ต้องการแก้ไข โปรแกรมจะแสดงข้อความ "ไม่พบห้องที่ต้องการแก้ไข"

8**. หากผู้ใช้เลือก '5' (ลบห้อง),** โปรแกรมจะขอเลขห้องที่ต้องการลบ และจากนั้นตรวจสอบว่าสามารถลบห้องได้หรือไม่ หากสามารถลบห้องได้ โปรแกรมจะแสดงข้อความ "ลบห้องสำเร็จ" ถ้าไม่พบห้องที่ต้องการลบ โปรแกรมจะแสดงข้อความ "ไม่พบห้องที่ต้องการลบ"

9. **หากผู้ใช้เลือก '6' (แสดงผลห้อง),** โปรแกรมจะแสดงเมนูทางหน้าจอเพื่อให้ผู้ใช้เลือกรูปแบบการแสดงผล โดยผู้ใช้สามารถเลือกแสดงทั้งหมด, ห้องที่ยังไม่มีคนจอง, ห้องที่มีคนจองแล้ว, หรือแสดงผลเพียงห้องที่ต้องการ โปรแกรมจะดำเนินการแสดงข้อมูลห้องตามที่ผู้ใช้เลือก

10. **หากผู้ใช้เลือก '7' (ค้นหาห้องตามประเภท),** โปรแกรมจะขอประเภทห้องที่ต้องการค้นหา และจากนั้นจะแสดงข้อมูลห้องที่มีประเภทตามที่ผู้ใช้เลือก

11. **หากผู้ใช้เลือก '8' (บันทึกข้อมูล),** โปรแกรมจะบันทึกข้อมูลห้องพักลงในไฟล์ CSV ที่เตรียมไว้

12**. หากผู้ใช้เลือก '9' (ออกจากโปรแกรม),** โปรแกรมจะแสดงข้อความ "ออกจากโปรแกรม" และจะจบการทำงานของโปรแกรม

13. หากผู้ใช้ป้อนตัวเลือกที่ไม่ถูกต้อง โปรแกรมจะแสดงข้อความ "กรุณาเลือกทำรายการใหม่"