**"Event Scheduler and Conflict Detector"**

def schedule\_event(events, event\_name, start\_time, end\_time):

"""Adds an event to the schedule, checking for conflicts."""

new\_event = {

'name': event\_name,

'start': start\_time,

'end': end\_time

}

conflicts = find\_conflicts(events, new\_event)

if conflicts:

print(f"Conflict detected: {event\_name} overlaps with {', '.join([e['name'] for e in conflicts])}")

suggested\_resolution = suggest\_resolution\_time(events, new\_event)

if suggested\_resolution:

print(f"Suggested resolution: Reschedule {event\_name} to {suggested\_resolution['start']} - {suggested\_resolution['end']}")

return False # Indicate failure to schedule due to conflict

else:

events.append(new\_event)

print(f"Event {event\_name} scheduled successfully.")

return True # Indicate successful scheduling

def find\_conflicts(events, new\_event):

"""Finds events that conflict with the new event."""

conflicts = []

for existing\_event in events:

if (new\_event['start'] < existing\_event['end'] and

new\_event['end'] > existing\_event['start']):

conflicts.append(existing\_event)

return conflicts

def suggest\_resolution\_time(events, new\_event):

"""Suggests an alternative time slot for the new event (basic implementation)."""

# For simplicity, this example just shifts the event by the duration of the conflict.

# In a real application, you'd want a more sophisticated approach.

conflict\_duration = 0

conflicts = find\_conflicts(events, new\_event)

if conflicts:

for existing\_event in conflicts:

overlap\_start = max(new\_event['start'], existing\_event['start'])

overlap\_end = min(new\_event['end'], existing\_event['end'])

conflict\_duration = max(conflict\_duration, overlap\_end - overlap\_start)

# Suggest shifting the new event's start time

suggested\_start = new\_event['end'] + conflict\_duration

suggested\_end = suggested\_start + (new\_event['end'] - new\_event['start'])

return {'start': suggested\_start, 'end': suggested\_end}

else:

return None # No conflict, no suggestion needed

# Example Usage:

events = []

schedule\_event(events, "Meeting A", 9, 10.5)

schedule\_event(events, "Workshop B", 10, 11.5) # Conflicts with Meeting A

schedule\_event(events, "Presentation C", 10.5, 12) # Conflicts with Workshop B

schedule\_event(events, "Lunch Break", 12, 13)

print("\nScheduled Events:")

for event in events:

print(f"{event['name']}: {event['start']} - {event['end']}")