```
# Colaborative Filter Recommendation System for Travel Agency using Lists and thier features

# List of itineraries
itineraries = [
    "Goa Beach Relaxation",
    "Himachal Trekking Adventure",
    "Rajasthan Desert Safari",
    "Kerala Backwater Houseboat",
    "Delhi Heritage Walk",
    "Madhya Pradesh Jungle Safari"
]

# Existing users' profiles (nested lists)
# Format: [places, activities, trip_types, budget_range, travel_frequency, travel_style, region_type]
user_profiles = [
    [["Goa", "Delhi"], ["Swimming", "Heritage Tour"], ["Family", "Friends"], "Medium", "6 months", "Repeat'
    [["Himachal", "Rajasthan"], ["Trekking", "Camel Ride"], ["Friends"], "Low", "1 year", "Explore", "Hilly
    [["Kerala", "MP"], ["Boating", "Wildlife"], ["Family"], "High", "3 months", "Repeat", "Forest"],
    [["Goa", "Himachal"], ["Trekking", "Swimming"], ["Friends", "Solo"], "Medium", "6 months", "Explore", '
]

user_names = ["A", "B", "C", "D"]

# Version 3 (improved)
# Function to get a new user's profile
def get_new_user_profile():
    print("\nPlease tell us about your travel preferences.")
    places = input("Places you've visited \n (Goa/ HImachal/ Rajasthan/ Kerala/ delhi/ MP) (comma separatec
    activities = input("Activities you enjoy (comma separated): ").title().split(",")

    # Optional removal of disliked activity
    dislike = input("Is there any activity you dislike and want to remove? (Optional): ").title().strip()
    if dislike and dislike in activities:
        activities.remove(dislike)  # list feature: .remove()

    trip_types = input("Type of trips you usually take (Family, Friends, Solo, Business): ").title().split(
    budget = input("Your usual budget? (Low / Medium / High): ").title()
    frequency = input("How often do you travel? (3 months / 6 months / 1 year): ").lower()
    style = input("Do you prefer similar trips or exploring new ones? (Repeat / Explore): ").title()
    region = input("Preferred region? (Coastal / Hilly / Forest / Metro): ").title()

    # Clean inputs using .strip()
    places = [p.strip() for p in places]
    activities = [a.strip() for a in activities]
    trip_types = [t.strip() for t in trip_types]

    # Sorting activities alphabetically
    activities.sort()  # list feature: .sort()

    profile = [places, activities, trip_types, budget, frequency]
    profile.insert(5, style)   # list feature: .insert()
    profile.insert(6, region)  # another insert

    return profile

# Similarity match score
def match_score(profile1, profile2):
    score = 0
    #list feature: list comprehension
    score += len([p for p in profile1[0] if p in profile2[0]])  # places
    score += len([a for a in profile1[1] if a in profile2[1]])  # activities
    score += len([t for t in profile1[2] if t in profile2[2]])  # trip types
    if profile1[3] == profile2[3]: score += 1  # budget
    if profile1[4] == profile2[4]: score += 1  # frequency
    if profile1[5] == profile2[5]: score += 1  # style
    if profile1[6] == profile2[6]: score += 1  # region
```

```
    return score

# Get new user profile
new_user = get_new_user_profile()

# Ask for user's name
new_user_name = input("Enter your name: ").strip().title()

# Handle name conflict using del, insert
if new_user_name in user_names:
    index = user_names.index(new_user_name)
    choice = input(f"A user with the name '{new_user_name}' already exists. Overwrite old record? (yes/no)
    if choice == "yes":
        del user_profiles[index]                # list feature: del
        del user_names[index]
        user_profiles.insert(index, new_user)   # list feature: .insert()
        user_names.insert(index, new_user_name)
    else:
        user_profiles.append(new_user)                      # list feature: .append()
        user_names.append(new_user_name + " (New)")         # show it's a duplicate
else:
    user_profiles.append(new_user)
    user_names.append(new_user_name)
```

```
Please tell us about your travel preferences.
Places you've visited
 (Goa/ HImachal/ Rajasthan/ Kerala/ delhi/ MP) (comma separated):  Delhi, Mumbai, Andaman
Activities you enjoy (comma separated):  Food Walks, Trekking
Is there any activity you dislike and want to remove? (Optional):
Type of trips you usually take (Family, Friends, Solo, Business):  Solo, friends
Your usual budget? (Low / Medium / High):  high
How often do you travel? (3 months / 6 months / 1 year):  3 months
Do you prefer similar trips or exploring new ones? (Repeat / Explore):  repeat
Preferred region? (Coastal / Hilly / Forest / Metro):  HIlly, Metro
Enter your name:  Alice
```

```
# Find best match user (excluding the new user themselves)
scores = []
for i in range(len(user_profiles) - 1):  # exclude the last user (new user)
    scores.append(match_score(new_user, user_profiles[i]))

best_match_index = scores.index(max(scores))  # list feature: .index()
matched_user = user_profiles[best_match_index]

print(f"\nInsight for Application Admin: This user matches most with User {user_names[best_match_index]} (

# Recommend itineraries based on matched user's data
recommendations = []

# Match ANY matching tag to suggest at least one itinerary
for place in matched_user[0]:
    if place == "Goa":
        recommendations.append("Goa Beach Relaxation")
    elif place == "Himachal":
        recommendations.append("Himachal Trekking Adventure")
    elif place == "Rajasthan":
        recommendations.append("Rajasthan Desert Safari")
    elif place == "Kerala":
        recommendations.append("Kerala Backwater Houseboat")
    elif place == "Delhi":
        recommendations.append("Delhi Heritage Walk")
    elif place == "MP":
        recommendations.append("Madhya Pradesh Jungle Safari")

for activity in matched_user[1]:
    if activity == "Swimming" and "Goa Beach Relaxation" not in recommendations:
        recommendations.append("Goa Beach Relaxation")
    elif activity == "Trekking" and "Himachal Trekking Adventure" not in recommendations:
        recommendations.append("Himachal Trekking Adventure")
    elif activity == "Camel Ride" and "Rajasthan Desert Safari" not in recommendations:
```

```
            recommendations.append("Rajasthan Desert Safari")
        elif activity == "Boating" and "Kerala Backwater Houseboat" not in recommendations:
            recommendations.append("Kerala Backwater Houseboat")
        elif activity == "Heritage Tour" and "Delhi Heritage Walk" not in recommendations:
            recommendations.append("Delhi Heritage Walk")
        elif activity == "Wildlife" and "Madhya Pradesh Jungle Safari" not in recommendations:
            recommendations.append("Madhya Pradesh Jungle Safari")

    # Remove duplicates
    recommendations = list(set(recommendations))  # list feature: set + list

    # Sort recommendations
    recommendations = sorted(recommendations)  # list feature: sorted()

    # Output recommendations
    print("\nTop Recommended Itineraries for You:")
    if recommendations:
        for r in recommendations:
            print("- " + r)
    else:
        print("We found partial matches, but couldn’t generate exact itineraries — please check back with more

    # Print updated profile database
    print("\n--- Updated User Database ---")
    for i in range(len(user_profiles)):
        print(f"User {user_names[i]}: {user_profiles[i]}")
```

```
Insight for Application Admin: This user matches most with User Scarlet (Score: 4)

Top Recommended Itineraries for You:
- Himachal Trekking Adventure

--- Updated User Database ---
User A: [['Goa', 'Delhi'], ['Swimming', 'Heritage Tour'], ['Family', 'Friends'], 'Medium', '6 months', 'Repeat', 'Coastal']
User B: [['Himachal', 'Rajasthan'], ['Trekking', 'Camel Ride'], ['Friends'], 'Low', '1 year', 'Explore', 'Hilly']
User C: [['Kerala', 'MP'], ['Boating', 'Wildlife'], ['Family'], 'High', '3 months', 'Repeat', 'Forest']
User D: [['Goa', 'Himachal'], ['Trekking', 'Swimming'], ['Friends', 'Solo'], 'Medium', '6 months', 'Explore', 'Hilly']
User Emily: [['Goa', 'Delhi'], ['Heritage Tour', 'Swimming'], ['Family', 'Friends'], 'Medium', '6 months', 'Repeat', 'Coastal']
User Scarlet: [['Darjeeling', 'Andaman'], ['Scuba Diving', 'Trekking'], ['Friends', 'Solo'], 'Medium', '6 months', 'Explore', 'Hilly
User Alice: [['Delhi', 'Mumbai', 'Andaman'], ['Food Walks', 'Trekking'], ['Solo', 'Friends'], 'High', '3 months', 'Repeat', 'Hilly,
```

```
#2nd version below (unimproved version of the above code)


# Function to get a new user's profile
def get_new_user_profile():
    print("\nPlease tell us about your travel preferences.")
    places = input("Places you’ve visited (comma separated): ").title().split(",")
    activities = input("Activities you enjoy (comma separated): ").title().split(",")

    # Optional removal of disliked activity
    dislike = input("Is there any activity you dislike and want to remove? (Optional): ").title().strip()
    if dislike and dislike in activities:
        activities.remove(dislike)  # list feature: .remove()

    trip_types = input("Type of trips you usually take (Family, Friends, Solo, Business): ").title().split
    budget = input("Your usual budget? (Low / Medium / High): ").title()
    frequency = input("How often do you travel? (3 months / 6 months / 1 year): ").lower()
    style = input("Do you prefer similar trips or exploring new ones? (Repeat / Explore): ").title()
    region = input("Preferred region? (Coastal / Hilly / Forest / Metro): ").title()

    # Clean inputs using .strip()
    places = [p.strip() for p in places]
    activities = [a.strip() for a in activities]
    trip_types = [t.strip() for t in trip_types]

    # Sorting activities alphabetically
    activities.sort()  # list feature: .sort()

    profile = [places, activities, trip_types, budget, frequency]
```

```
        profile.insert(5, style)   # list feature: .insert()
        profile.insert(6, region)  # another insert

        return profile

    # Similarity match score
    def match_score(profile1, profile2):
        score = 0
        score += len([p for p in profile1[0] if p in profile2[0]])  # places
        score += len([a for a in profile1[1] if a in profile2[1]])  # activities
        score += len([t for t in profile1[2] if t in profile2[2]])  # trip types
        if profile1[3] == profile2[3]: score += 1  # budget
        if profile1[4] == profile2[4]: score += 1  # frequency
        if profile1[5] == profile2[5]: score += 1  # style
        if profile1[6] == profile2[6]: score += 1  # region
        return score

    # Get new user profile
    new_user = get_new_user_profile()

    # Ask for user's name
    new_user_name = input("Enter your name: ").strip().title()

    # Handle name conflict using del, insert
    if new_user_name in user_names:
        index = user_names.index(new_user_name)
        choice = input(f"A user with the name '{new_user_name}' already exists. Overwrite old record? (yes/no)
        if choice == "yes":
            del user_profiles[index]                # list feature: del
            del user_names[index]
            user_profiles.insert(index, new_user)   # list feature: .insert()
            user_names.insert(index, new_user_name)
        else:
            user_profiles.append(new_user)                      # list feature: .append()
            user_names.append(new_user_name + " (New)")         # show it's a duplicate
    else:
        user_profiles.append(new_user)
        user_names.append(new_user_name)

    # Find best match user
    scores = []
    for profile in user_profiles:
        scores.append(match_score(new_user, profile))

    best_match_index = scores.index(max(scores))  # list feature: .index()

    print(f"\nInsight for Application Admin: This user match most with User {user_names[best_match_index]} (Sc

    # Recommend itineraries based on matched user's places & activities
    matched_user = user_profiles[best_match_index]
    recommendations = []

    if "Goa" in matched_user[0] and "Swimming" in matched_user[1]:
        recommendations += ["Goa Beach Relaxation"]  # list feature: +
    if "Himachal" in matched_user[0] and "Trekking" in matched_user[1]:
        recommendations += ["Himachal Trekking Adventure"]
    if "Rajasthan" in matched_user[0] and "Camel Ride" in matched_user[1]:
        recommendations += ["Rajasthan Desert Safari"]
    if "Kerala" in matched_user[0] and "Boating" in matched_user[1]:
        recommendations += ["Kerala Backwater Houseboat"]
    if "Delhi" in matched_user[0] and "Heritage Tour" in matched_user[1]:
        recommendations += ["Delhi Heritage Walk"]
    if "MP" in matched_user[0] and "Wildlife" in matched_user[1]:
        recommendations += ["Madhya Pradesh Jungle Safari"]

    # sort recommendations
    recommendations = sorted(recommendations)  # list feature: sorted()
```

```
# Output recommendations
print("\nTop Recommended Itineraries for You:")
if recommendations:
    for r in recommendations:
        print("- " + r)
else:
    print("No matching itineraries found.")

# Print updated profile database
print("\n--- Updated User Database ---")
for i in range(len(user_profiles)):
    print(f"User {user_names[i]}: {user_profiles[i]}")
```

```
Please tell us about your travel preferences.
Places you've visited (comma separated):  Darjeeling, Andaman
Activities you enjoy (comma separated):  Trekking, Scuba diving
Is there any activity you dislike and want to remove? (Optional):
Type of trips you usually take (Family, Friends, Solo, Business):  Friends, Solo
Your usual budget? (Low / Medium / High):  Medium
How often do you travel? (3 months / 6 months / 1 year):  6 months
Do you prefer similar trips or exploring new ones? (Repeat / Explore):  Explore
Preferred region? (Coastal / Hilly / Forest / Metro):  Hilly, island
Enter your name:  Scarlet

Insight for Application Admin: This user match most with User Scarlet (Score: 10)

Top Recommended Itineraries for You:
No matching itineraries found.

--- Updated User Database ---
User A: [['Goa', 'Delhi'], ['Swimming', 'Heritage Tour'], ['Family', 'Friends'], 'Medium', '6 months', 'Repeat', 'Coastal']
User B: [['Himachal', 'Rajasthan'], ['Trekking', 'Camel Ride'], ['Friends'], 'Low', '1 year', 'Explore', 'Hilly']
User C: [['Kerala', 'MP'], ['Boating', 'Wildlife'], ['Family'], 'High', '3 months', 'Repeat', 'Forest']
User D: [['Goa', 'Himachal'], ['Trekking', 'Swimming'], ['Friends', 'Solo'], 'Medium', '6 months', 'Explore', 'Hilly']
User Emily: [['Goa', 'Delhi'], ['Heritage Tour', 'Swimming'], ['Family', 'Friends'], 'Medium', '6 months', 'Repeat', 'Coastal']
User Scarlet: [['Darjeeling', 'Andaman'], ['Scuba Diving', 'Trekking'], ['Friends', 'Solo'], 'Medium', '6 months', 'Explore', 'Hilly
```

```
# Find best match user (excluding the new user themselves)
scores = []
for i in range(len(user_profiles) - 1):  # exclude the last user (new user)
    scores.append(match_score(new_user, user_profiles[i]))

best_match_index = scores.index(max(scores))  # list feature: .index()
matched_user = user_profiles[best_match_index]

print(f"\nInsight for Application Admin: This user matches most with User {user_names[best_match_index]}

# Recommend itineraries based on matched user's data
recommendations = []

# Match ANY matching tag to suggest at least one itinerary
for place in matched_user[0]:
    if place == "Goa":
        recommendations.append("Goa Beach Relaxation")
    elif place == "Himachal":
        recommendations.append("Himachal Trekking Adventure")
    elif place == "Rajasthan":
        recommendations.append("Rajasthan Desert Safari")
    elif place == "Kerala":
        recommendations.append("Kerala Backwater Houseboat")
    elif place == "Delhi":
        recommendations.append("Delhi Heritage Walk")
    elif place == "MP":
        recommendations.append("Madhya Pradesh Jungle Safari")

for activity in matched_user[1]:
    if activity == "Swimming" and "Goa Beach Relaxation" not in recommendations:
        recommendations.append("Goa Beach Relaxation")
    elif activity == "Trekking" and "Himachal Trekking Adventure" not in recommendations:
        recommendations.append("Himachal Trekking Adventure")
    elif activity == "Camel Ride" and "Rajasthan Desert Safari" not in recommendations:
```

```
            recommendations.append("Rajasthan Desert Safari")
        elif activity == "Boating" and "Kerala Backwater Houseboat" not in recommendations:
            recommendations.append("Kerala Backwater Houseboat")
        elif activity == "Heritage Tour" and "Delhi Heritage Walk" not in recommendations:
            recommendations.append("Delhi Heritage Walk")
        elif activity == "Wildlife" and "Madhya Pradesh Jungle Safari" not in recommendations:
            recommendations.append("Madhya Pradesh Jungle Safari")

    # Remove duplicates
    recommendations = list(set(recommendations))  # list feature: set + list

    # Sort recommendations
    recommendations = sorted(recommendations)  # list feature: sorted()

    # Output recommendations
    print("\nTop Recommended Itineraries for You:")
    if recommendations:
        for r in recommendations:
            print("- " + r)
    else:
        print("We found partial matches, but couldn’t generate exact itineraries — please check back with more

    # Print updated profile database
    print("\n--- Updated User Database ---")
    for i in range(len(user_profiles)):
        print(f"User {user_names[i]}: {user_profiles[i]}")
```

```
Insight for Application Admin: This user matches most with User D (Score: 6)

Top Recommended Itineraries for You:
- Goa Beach Relaxation
- Himachal Trekking Adventure

--- Updated User Database ---
User A: [['Goa', 'Delhi'], ['Swimming', 'Heritage Tour'], ['Family', 'Friends'], 'Medium', '6 months', 'Repeat', 'Coastal']
User B: [['Himachal', 'Rajasthan'], ['Trekking', 'Camel Ride'], ['Friends'], 'Low', '1 year', 'Explore', 'Hilly']
User C: [['Kerala', 'MP'], ['Boating', 'Wildlife'], ['Family'], 'High', '3 months', 'Repeat', 'Forest']
User D: [['Goa', 'Himachal'], ['Trekking', 'Swimming'], ['Friends', 'Solo'], 'Medium', '6 months', 'Explore', 'Hilly']
User Emily: [['Goa', 'Delhi'], ['Heritage Tour', 'Swimming'], ['Family', 'Friends'], 'Medium', '6 months', 'Repeat', 'Coastal']
User Scarlet: [['Darjeeling', 'Andaman'], ['Scuba Diving', 'Trekking'], ['Friends', 'Solo'], 'Medium', '6 months', 'Explore', 'Hilly
```