Additional Set-Based Python Programs

1. Remove duplicate email addresses from a mailing list

```
emails = ["a@example.com", "b@example.com", "a@example.com"]
unique_emails = set(emails)
print("Unique emails:", unique_emails)
```

2. Track unique website visitors using set of IPs

```
ips = {"192.168.1.1", "192.168.1.2", "192.168.1.1"}
print("Unique visitors:", len(ips))
```

3. Find common users between two product sales using sets

```
product1_users = {"user1", "user2", "user3"}
product2_users = {"user2", "user4"}
common_users = product1_users & product2_users
print("Common users:", common_users)
```

4. Find items common in two festival shopping lists

```
list1 = {"lights", "sweets", "crackers"}
list2 = {"crackers", "sweets", "gifts"}
common_items = list1 & list2
print("Common items:", common_items)
```

5. Check if a winning ticket number is in the participant set

```
participants = {"T001", "T002", "T003"}
winning_ticket = "T002"
print("Winner found:" if winning_ticket in participants else "Not a winner")
```

6. Check which candidate skills match job requirements

```
candidate_skills = {"Python", "SQL", "Excel"}
job_requirements = {"Python", "SQL"}
matches = job_requirements <= candidate_skills
print("Meets requirements:" if matches else "Does not meet requirements")
```

7. Find common symptoms among multiple patients

```
patient1 = {"fever", "cough", "headache"}
```

```
patient2 = {"cough", "fever", "fatigue"}
common_symptoms = patient1 & patient2
print("Common symptoms:", common_symptoms)
```

8. Suggest mutual friends using intersection of sets

```
friends_A = {"Ravi", "Priya", "Aman"}
friends_B = {"Aman", "Neha", "Priya"}
mutual_friends = friends_A & friends_B
print("Mutual friends:", mutual_friends)
```

9. Check which products are missing from stock

```
required_products = {"Rice", "Flour", "Oil", "Sugar"}
in_stock = {"Rice", "Sugar"}
missing = required_products - in_stock
print("Missing products:", missing)
```

10. Find who responded to both or only one of two surveys

```
survey1 = {"A", "B", "C"}
survey2 = {"B", "C", "D"}
both = survey1 & survey2
only_one = survey1 ^ survey2
print("Responded to both:", both)
print("Responded to only one:", only_one)
```

11. Check if a directory contains duplicate files

```
files = ["report.doc", "data.csv", "report.doc"]
duplicates = len(files) != len(set(files))
print("Duplicates found" if duplicates else "No duplicates")
```

12. Get users interested in both music and sports

```
music_lovers = {"A", "B", "C"}
sports_lovers = {"B", "C", "D"}
interested_in_both = music_lovers & sports_lovers
print("Users interested in both:", interested_in_both)
```