NAME:TINA BORUNDIA

ROLL NO:16

PRACTICLE :3

1] Write a program to create a set containing 10 randomly generated numbers in the range 15 to 45. Count how many of these numbers are less than 30. Delete all numbers which are greater than 35.

In [ ]:
```
import random
my_set=set({})

for i in range(10):
    my_set.add(random.randrange(15,45))
print(my_set)
```

```
{35, 38, 39, 15, 20, 23, 25, 29}
```

In [ ]:
```
count=0
for i in my_set:
    if i<30:
        count+=1
print(count)
```

```
5
```

In [ ]:
```
for i in my_set.copy():
    if i>35 :
        my_set.remove(i)
print(my_set)
```

```
{35, 15, 20, 23, 25, 29}
```

2] In a survey of 500 students of a college, it was found that 49% liked watching football, 53% liked watching hockey and 62% liked watching basketball. Also, 27% liked watching football and hockey both, 29% liked watching basketball and hockey both and 28% liked watching football and basketball both. 5% liked watching none of these games.

In [ ]:
```
total_students = 500
per_football = 49
per_hockey = 53
per_basketball = 62
per_fh = 27
per_bh = 29
per_fb = 28
per_none = 5

F = (per_football / 100) * total_students
H = (per_hockey / 100) * total_students
B = (per_basketball / 100) * total_students
FH = (per_fh / 100) * total_students
BH = (per_bh / 100) * total_students
FB = (per_fb / 100) * total_students
N = (per_none / 100) * total_students
### 75
### 0.75
### 205
### 270
```

i. How many students like watching all the three games?

In [ ]:
```
All = F + H + B - (FH + BH + FB) + N
print("Number of students who like all three games:", int(All))
```

```
Number of students who like all three games: 425
```

ii. Find the ratio of number of students who like watching only football to those who like watching only hockey.

In [ ]:
```
OnlyFootball = F - FH - FB
OnlyHockey = H - FH - BH
ratio_football_to_hockey = OnlyFootball / OnlyHockey
print("Ratio of students who like only football to those who like only hocke
y:", round(ratio_football_to_hockey, 2))
```

```
Ratio of students who like only football to those who like only hockey: 2.0
```

iii. Find the number of students who like watching only one of the three given games.

In [ ]:
```
OnlyOneGame = OnlyFootball + OnlyHockey + (B - BH - FB)
print("Number of students who like watching only one of the three games:", int
(OnlyOneGame))
```

```
Number of students who like watching only one of the three games: -20
```

iv. Find the number of students who like watching at least two of the given games.

In [ ]:
```
AtLeastTwoGames = FH + BH + FB
print("Number of students who like watching at least two of the given games:",
int(AtLeastTwoGames))
```

```
Number of students who like watching at least two of the given games: 420
```

In [ ]:
```
### 3} For the purposes of marketing research, a survey of some men is conduct
ed in a town.  The results shows that Set C liked watching comedies, Set F lik
ed watching fantasy movies and Set R liked watching romantic movies
C={203, 204, 207, 216, 217, 219, 226, 227, 230, 233, 239, 248, 250, 254, 260,
262, 272, 279, 281, 282, 289, 291, 292, 293, 294, 298, 299, 304, 307, 308, 30
9,
315, 322, 324, 331, 332, 337, 343, 345, 346, 347, 348, 349, 350, 354, 358, 36
1,
368, 370, 371, 373, 374, 376, 379, 383, 385, 388, 392, 393, 395, 397, 406, 40
9,
412, 414, 416, 417, 420, 421, 427, 433, 444, 446, 448, 454, 455, 456, 457, 45
9,
461, 467, 469, 470, 473, 482, 487, 494, 499}
F={200, 202, 203, 220, 222, 225, 226, 229, 237, 241, 252, 253, 260, 262, 264,
269, 270, 276, 282, 283, 289, 290, 293, 296, 298, 300, 302, 309, 311, 312, 31
3,
319, 320, 322, 325, 328, 329, 337, 342, 343, 353, 355, 357, 358, 363, 364, 37
0,
371, 374, 375, 381, 382, 390, 395, 400, 406, 415, 416, 418, 420, 421, 428, 43
2,
435, 436, 437, 438, 444, 452, 458, 461, 471, 472, 476, 478, 479, 484, 488, 49
0,
494, 497, 498, 499}
R={203, 204, 205, 207, 211, 212, 213, 220, 223, 225, 226, 229, 233, 235, 244,
245, 246, 247, 248, 251, 252, 254, 256, 259, 261, 265, 267, 269, 270, 281, 28
3,
287, 289, 290, 295, 298, 302, 324, 332, 335, 336, 338, 344, 349, 350, 355, 35
6,

359, 362, 364, 367, 369, 374, 375, 377, 378, 382, 384, 392, 394, 395, 398, 40
1,
403, 406, 407, 410, 414, 416, 424, 426, 429, 431, 432, 435, 439, 442, 454, 46
0,
461, 462, 466, 467, 472, 473, 476, 496, 498}
### Sets contains men's registration numbers.
### i. How many women like watching all the three movie genres?
### ii. Find the number of women who like watching only one of the three genre
s.
### iii. Find the number of women who like watching at least two of the given
genres.
```

In [ ]:
```
C = {203, 204, 207, 216, 217, 219, 226, 227, 230, 233, 239, 248, 250, 254, 26
0,
     262, 272, 279, 281, 282, 289, 291, 292, 293, 294, 298, 299, 304, 307, 30
8,
     309, 315, 322, 324, 331, 332, 337, 343, 345, 346, 347, 348, 349, 350, 35
4,
     358, 361, 368, 370, 371, 373, 374, 376, 379, 383, 385, 388, 392, 393, 39
5,
     397, 406, 409, 412, 414, 416, 417, 420, 421, 427, 433, 444, 446, 448, 45
4,
     455, 456, 457, 459, 461, 467, 469, 470, 473, 482, 487, 494, 499}
F = {200, 202, 203, 220, 222, 225, 226, 229, 237, 241, 252, 253, 260, 262, 26
4,
     269, 270, 276, 282, 283, 289, 290, 293, 296, 298, 300, 302, 309, 311, 31
2,
     313, 319, 320, 322, 325, 328, 329, 337, 342, 343, 353, 355, 357, 358, 36
3,
     364, 370, 371, 374, 375, 381, 382, 390, 395, 400, 406, 415, 416, 418, 42
0,
     421, 428, 432, 435, 436, 437, 438, 444, 452, 458, 461, 471, 472, 476, 47
8,
     479, 484, 488, 490, 494, 497, 498, 499}
R = {203, 204, 205, 207, 211, 212, 213, 220, 223, 225, 226, 229, 233, 235, 24
4,
     245, 246, 247, 248, 251, 252, 254, 256, 259, 261, 265, 267, 269, 270, 28
1,
     283, 287, 289, 290, 295, 298, 302, 324, 332, 335, 336, 338, 344, 349, 35
0,
     355, 356, 359, 362, 364, 367, 369, 374, 375, 377, 378, 382, 384, 392, 39
4,
     395, 398, 401, 403, 406, 407, 410, 414, 416, 424, 426, 429, 431, 432, 43
5,
     439, 442, 454, 460, 461, 462, 466, 467, 472, 473, 476, 496, 498}

like_all_three_genres = len(C.intersection(F).intersection(R))
like_only_one_genre = len((C - F - R).union(F - C - R).union(R - C - F))
like_at_least_two_genres = len((C.intersection(F)).union(C.intersection(R)).un
ion(F.intersection(R)))

print("i. Number who like all three movie genres:", like_all_three_genres)
print("ii. Number who like watching only one of the three genres:", like_only_
one_genre)
print("iii. Number who like watching at least two of the given genres:", like_
at_least_two_genres)
```

```
i. Number who like all three movie genres: 9
ii. Number who like watching only one of the three genres: 134
iii. Number who like watching at least two of the given genres: 58
```

4] Write a program to input your friend's names and their Phone Numbers and store them in the dictionary as the key-value pair. Perform the following operations on the dictionary:

i. Display the name and phone number of all your friends

ii. Add a new key-value pair in this dictionary and display the modified dictionary

iii. Delete a particular friend from the dictionary

iv. Modify the phone number of an existing friend

v. Check if a friend is present in the dictionary or not

vi. Display the dictionary in sorted order of names

In [1]:
```
from collections import OrderedDict
n=int(input("Enter number of friends"))
dict1={}
for i in range(n):
 name=input("Enter your name")
 phone_number=int(input("Enter phone number"))
 dict1[name]=phone_number
print(dict1) #i
#ii
dict1['Aishu']=8387939822
dict1
# #iii
dict1.pop('Aishu')
dict1
#iv
dict1['Priya']=87479449343
print(dict1)
#v
if 'Priya' in dict1:
 print("Got it")
else:
 print("No")
#vi
# sorted_dict1={}
# sorted_dict1=sorted(dict1.items())
# dict1=dict(sorted_dict1)
# print(dict1)
```

```
Enter number of friends3
Enter your nameTina
Enter phone number673493974
Enter your namesiya
Enter phone number739738742
Enter your nameRiya
Enter phone number83767648
{'Tina': 673493974, 'siya': 739738742, 'Riya': 83767648}
{'Tina': 673493974, 'siya': 739738742, 'Riya': 83767648, 'Priya': 8747944934
3}
Got it
```

5] Create a nested dictionary as given below. Food_blogger= { City_name: {food_speciality: [outlet1,outlet2,….outletn]}}

Example:Food_blogger= { “Nagpur” : { “poha”: [“Bardi”, “KP ground”] } “Nasik”: { “samosa”: [“ABC”, “XYZ”] }}

When you run your program, it should

i. ask the user to enter a city name (at least 5), and return the details of food specialties (at least 2 for each city) of that city back to the user.

ii. ask the user to enter a food specialty and return the name of that city back to the user.

iii. ask the user to enter a city and food specialty and return the name of outlets back to the user.

In [ ]:
```
food_blogger = {
    "Nagpur":
    {
        "poha": ["Bardi", "KP ground"],
        "tarri poha": ["Sita Buldi", "Sadar"],
    },
    "Nasik":
    {
        "samosa": ["Gandhi Nagar", "Marwadi chowk"],
        "misal pav": ["Panchavati", "Balaji Chowk"],
    },
    "Mumbai":
    {
        "vada pav": ["Ashok", "Shivaji"],
        "pav bhaji": ["Sadar", "Balaji"],
    },
    "Pune":
    {
        "misal pav": ["Bedekar", "Katariya"],
        "chaat": ["Aaswad", "Chaitanya"],
    },
    "Delhi":
    {
        "chaat": ["Chandni Chowk", "Karol Bagh"],
        "pulao": ["Jain's", "Kothari's"],
    }
}
```

In [ ]:
```
city_name = input("Enter a city name: ")
if city_name in food_blogger:
    print(f"Food specialties in {city_name}:")
    for food_specialty in food_blogger[city_name]:
        print(food_specialty)
else:
    print("City not found.")
```

```
Enter a city name: Delhi
Food specialties in Delhi:
chaat
pulao
```

In [ ]:
```
food_specialty = input("Enter a food specialty: ")
found = False
for city, specialties in food_blogger.items():
    if food_specialty in specialties:
        print(f"The food specialty '{food_specialty}' is available in {cit
y}.")
        found = True
if not found:
    print("Food specialty not found in any city.")
```

```
Enter a food specialty: pav bhaji
The food specialty 'pav bhaji' is available in Mumbai.
```

In [ ]:
```
city_name = input("Enter a city name: ")
food_specialty = input("Enter a food specialty: ")
if city_name in food_blogger and food_specialty in food_blogger[city_name]:
    print(f"Outlets for {food_specialty} in {city_name}:")
    for outlet in food_blogger[city_name][food_specialty]:
        print(outlet)
else:
    print("City or food specialty not found.")
```

```
Enter a city name: Pune
Enter a food specialty: chaat
Outlets for chaat in Pune:
Aaswad
Chaitanya
```

6. Consider the information given below and answer the following question.

Employee_data = {101:['Shiva', 24, 'Content Strategist'] ,102:['Udit',25,'CoStrategist'], 103:[Sonam, 28,Sr Manager], 104:['Ansari',29,'Product Lead'],105:[Huzefa,32,Project Manager&]}

i. Get details of the oldest Employee

ii. Identify the age of the employee with employee id 159 [If the employee isn't present return NA]

iii. Count the total number of employees in the organization

iv. Calculate the mean age of the employees

v. Update the ages of employee id - 104,140, and 164 as 27 and then calculate the updated mean age of the employees

In [13]:
```
Employee_data = {101: ['Shiva', 24, 'Content Strategist'],
                 102: ['Udit', 25, 'CoStrategist'],
                 103: ['Sonam', 28, 'Sr Manager'],
                 104: ['Ansari', 29, 'Product Lead'],
                 105: ['Huzefa', 32, 'Project Manager&']}

# i. Get details of the oldest Employee
oldest_employee = max(Employee_data.items(), key=lambda x: x[1][1])
oldest_employee_details = oldest_employee[1]
print("Details of the oldest Employee:", oldest_employee_details)

# ii. Identify the age of the employee with employee id 159 [ If the employee
isn't present return NA]
employee_id_to_check = 159
age_of_employee_159 = Employee_data.get(employee_id_to_check, 'NA')
print("Age of Employee with ID 159:", age_of_employee_159)

# iii. Count the total number of employees in the organization
total_employees = len(Employee_data)
print("Total number of employees:", total_employees)

# iv. Calculate the mean age of the employees
total_age = sum(employee[1][1] for employee in Employee_data.items())
mean_age = total_age / total_employees
print("Mean age of the employees:", mean_age)

# v. Update the ages of employee id - 104, 140, and 164 as 27
employee_ids_to_update = [104, 140, 164]
updated_mean_age_employees = []

for emp_id in employee_ids_to_update:
    if emp_id in Employee_data:
        Employee_data[emp_id][1] = 27
        updated_mean_age_employees.append(Employee_data[emp_id][1])

# Recalculate the mean age after the updates
updated_mean_age = sum(updated_mean_age_employees) / len(updated_mean_age_empl
oyees)
print("Updated mean age of the employees after age updates:", updated_mean_ag
e)
```

```
Details of the oldest Employee: ['Huzefa', 32, 'Project Manager&']
Age of Employee with ID 159: NA
Total number of employees: 5
Mean age of the employees: 27.6
Updated mean age of the employees after age updates: 27.0
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

1. Create a SORTED list of all values from the dictionary input_dict = {Jack Dorsey:Twitter , Tim Cook : Apple, Jeff Bezos : Amazon , Mukesh Ambani : RJIO} Sample Output: [Amazon, Apple, RJIO, Twitter]

In [ ]:
```
dict = {'Jack Dorsey' : 'Twitter' , 'Tim Cook' : 'Apple', 'Jeff Bezos' : 'Amaz
on' , 'Mukesh Ambani' : 'RJIO'}
print(dict)
```

```
{'Jack Dorsey': 'Twitter', 'Tim Cook': 'Apple', 'Jeff Bezos': 'Amazon', 'Muke
sh Ambani': 'RJIO'}
```

In [ ]:
```
li=[dict.get('Jack Dorsey'),dict.get('Tim Cook'),dict.get('Jeff Bezos'),dict.g
et('Mukesh Ambani') ]
li.sort()
print(li)
```

```
['Amazon', 'Apple', 'RJIO', 'Twitter']
```

In [ ]:
```
8. Create a Nested Dictionary Using the given table in the format:
   Olympic =
    {
       County1 :
       {
           Country Code-1 :
           {
               Gold : value , Silver : value , Bronze : value
           }
       },
       County2 :
       {
           Country Code-2 :
           {
               Gold : value , Silver : value , Bronze : value
           }
       }
   }
Country       || Country Code ||Year ||Medal-Gold|| Medal-Silver || Medal- Bro
nze
Great Britain || GBR          ||2012 || 29       || 17           || 19
China         || CHN          ||2012 || 38       || 28           || 22
Russia        || RUS          ||2012 || 24       || 25           || 32
United States || US           ||2012 || 46       || 28           || 29
Korea         || KOR          ||2012 || 13       || 8            || 7
Japan         || JPN          ||2012 || 7        || 14           || 17
Germany       || GER          ||2012 || 11       || 11           || 14

Write a Python Code to:
i.  Find the country with maximum gold medals
ii. Find the countries with more than 20 gold medals
iii.Evaluate the Dictionary and print the name of each country with its gold m
edals and total number of medals
```

In [9]:
```
# Step 1:
olympic_data = {
    "Great Britain": {"Country Code": "GBR", "2012": {"Gold": 29, "Silver": 1
7, "Bronze": 19}},
    "China": {"Country Code": "CHN", "2012": {"Gold": 38, "Silver": 28, "Bronz
e": 22}},
    "Russia": {"Country Code": "RUS", "2012": {"Gold": 24, "Silver": 25, "Bron
ze": 32}},
    "United States": {"Country Code": "US", "2012": {"Gold": 46, "Silver": 28,
"Bronze": 29}},
    "Korea": {"Country Code": "KOR", "2012": {"Gold": 13, "Silver": 8, "Bronz
e": 7}},
    "Japan": {"Country Code": "JPN", "2012": {"Gold": 7, "Silver": 14, "Bronz
e": 17}},
    "Germany": {"Country Code": "GER", "2012": {"Gold": 11, "Silver": 11, "Bro
nze": 14}}
}
```

In [10]:
```
# Step 2:
max_gold_country = max(olympic_data, key=lambda k: olympic_data[k]["2012"]["Go
ld"])

# Step 3:
more_than_20_gold_countries = [country for country in olympic_data if olympic_
data[country]["2012"]["Gold"] > 20]

# Step 4:
for country, data in olympic_data.items():
    gold_medals = data["2012"]["Gold"]
    total_medals = sum(data["2012"].values())
    print(f"{country} - Gold Medals: {gold_medals}, Total Medals: {total_medal
s}")

print("\nCountry with Maximum Gold Medals:", max_gold_country)
print("Countries with more than 20 Gold Medals:", more_than_20_gold_countries)
```

```
Great Britain - Gold Medals: 29, Total Medals: 65
China - Gold Medals: 38, Total Medals: 88
Russia - Gold Medals: 24, Total Medals: 81
United States - Gold Medals: 46, Total Medals: 103
Korea - Gold Medals: 13, Total Medals: 28
Japan - Gold Medals: 7, Total Medals: 38
Germany - Gold Medals: 11, Total Medals: 36

Country with Maximum Gold Medals: United States
Countries with more than 20 Gold Medals: ['Great Britain', 'China', 'Russia',
'United States']
```

1. Take the string your name as input (Ex. Firstname middlename surname). Generate the mail id as surname followed by first letter from firstname, first letter from middle name, @rknec.edu and store this into the dictionary as {Mail ID: Name} using function. Do it for 5 students and store it in the dictionary. Eg. Input : Sachin Ramesh Tendulkar (value) Mail id: tendulkarsr@rknec.edu (key)

In [12]:
```
email_id_dict = {}

for _ in range(5):
    full_name = input("Enter the full name (e.g., Firstname Middlename Surnam
e): ")

    names = full_name.split()
    first_name = names[0]
    middle_name = names[1][0] if len(names) > 2 else ""
    surname = names[-1]

    email_id = f"{surname.lower()}{first_name[0].lower()}{middle_name.lower()}
@rknec.edu"

    email_id_dict[email_id] = full_name
print("\nDictionary - {Mail ID: Name}:")
for email_id, name in email_id_dict.items():
    print(f"{email_id}: {name}")
```

```
Enter the full name (e.g., Firstname Middlename Surname): Tina Pradip Borundi
a
Enter the full name (e.g., Firstname Middlename Surname): Riya Satish Satpute
Enter the full name (e.g., Firstname Middlename Surname): Sanika Ramesh Mandl
e
Enter the full name (e.g., Firstname Middlename Surname): Shreya Krishnarao S
hinde
Enter the full name (e.g., Firstname Middlename Surname): Dhawal Anand Borund
ia

Dictionary - {Mail ID: Name}:
borundiatp@rknec.edu: Tina Pradip Borundia
satputers@rknec.edu: Riya Satish Satpute
mandlesr@rknec.edu: Sanika Ramesh Mandle
shindesk@rknec.edu: Shreya Krishnarao Shinde
borundiada@rknec.edu: Dhawal Anand Borundia
```

In [ ]:

In [ ]: