ICP-1 Report

```
[2]  # Mount Google Drive
     from google.colab import drive
     drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

Generate | a slider using jupyter widgets

```
[3]  # Take input from the user
     input_str = input("Enter a string: ")
     # Delete at least 2 characters
     modified_str = input_str[:-2]
     # Reverse the modified string
     reversed_str = modified_str[::-1]

     # Print the reversed string
     print("Result:", reversed_str)
```

```
Enter a string: python
Result: htyp
```

```
[4]  # Take input from the user
     num1 = float(input("Enter the first number: "))
     num2 = float(input("Enter the second number: "))

     # Perform arithmetic operations
     add = num1 + num2
     sub = num1 - num2
     mul = num1 * num2

     # Ensure non-zero division
     if num2 != 0:
         div = num1 / num2
     else:
         div = "Cannot divide by zero"

     # Print the results
     print("Addition:", add)
     print("Subtraction:", sub)
     print("Multiplication:", mul)
     print("Division:", div)
```

```
Enter the first number: 3
Enter the second number: 4
Addition: 7.0
Subtraction: -1.0
Multiplication: 12.0
Division: 0.75
```

[5] 14s

```python
# Take input from the user
input_sentence = input("Enter a sentence: ")

# Replace 'python' with 'pythons'
output_sentence = input_sentence.replace('python', 'pythons')

# Print the modified sentence
print("Modified sentence:", output_sentence)
```

```
Enter a sentence: i love playing with python
Modified sentence: i love playing with pythons
```

[6] 36s

```python
# Take input from the user
cs = float(input("Enter the class score: "))

# Determine the letter grade based on the score
if cs >= 90:
    letter_grade = 'A'
elif 80 <= cs < 90:
    letter_grade = 'B'
elif 70 <= cs < 80:
    letter_grade = 'C'
elif 60 <= cs < 70:
    letter_grade = 'D'
else:
    letter_grade = 'F'

# Print the letter grade
print("Letter Grade:", letter_grade)
```

```
Enter the class score: 90
Letter Grade: A
```

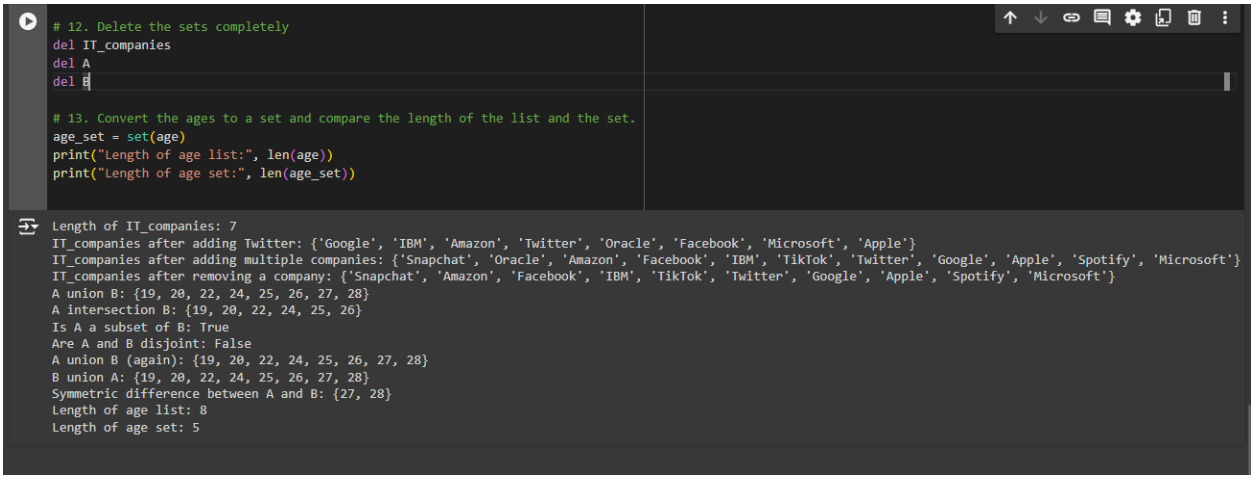
0s

```python
# Input list
x = [23, 'Python', 23.98]

# Output the original list
print(x)

# Create a new list containing the types of each element in x
types_list = [type(item) for item in x]

# Output the types list
print(types_list)
```

```
[23, 'Python', 23.98]
[<class 'int'>, <class 'str'>, <class 'float'>]
```

```python
# Given sets and list
IT_companies = {'Facebook', 'Google', 'Microsoft', 'Apple', 'IBM', 'Oracle', 'Amazon'}
A = {19, 22, 24, 20, 25, 26}
B = {19, 22, 20, 25, 26, 24, 28, 27}
age = [22, 19, 24, 25, 26, 24, 25, 24]

# 1. Find the length of the set IT_companies
length_IT_companies = len(IT_companies)
print("Length of IT_companies:", length_IT_companies)

# 2. Add 'Twitter' to IT_companies
IT_companies.add('Twitter')
print("IT_companies after adding Twitter:", IT_companies)

# 3. Insert multiple IT companies at once to the set IT_companies
IT_companies.update(['Snapchat', 'TikTok', 'Spotify'])
print("IT_companies after adding multiple companies:", IT_companies)

# 4. Remove one of the companies from the set IT_companies
IT_companies.remove('Oracle')  # Example of removal
print("IT_companies after removing a company:", IT_companies)

# 5. Difference between remove and discard:
# - remove() will raise a KeyError if the element is not present in the set.- discard() will not raise an error if the element is not present in the set.
```

```python
# 6. Join A and B
A_union_B = A.union(B)
print("A union B:", A_union_B)

# 7. Find A intersection B
A_intersection_B = A.intersection(B)
print("A intersection B:", A_intersection_B)

# 8. Is A subset of B
is_A_subset_B = A.issubset(B)
print("Is A a subset of B:", is_A_subset_B)

# 9. Are A and B disjoint sets
are_A_B_disjoint = A.isdisjoint(B)
print("Are A and B disjoint:", are_A_B_disjoint)

# 10. Join A with B and B with A
A_union_B_again = A.union(B)
B_union_A = B.union(A)
print("A union B (again):", A_union_B_again)
print("B union A:", B_union_A)

# 11. What is the symmetric difference between A and B
A_symmetric_difference_B = A.symmetric_difference(B)
print("Symmetric difference between A and B:", A_symmetric_difference_B)
```



```
# 12. Delete the sets completely
del IT_companies
del A
del B

# 13. Convert the ages to a set and compare the length of the list and the set.
age_set = set(age)
print("Length of age list:", len(age))
print("Length of age set:", len(age_set))
```

```
Length of IT_companies: 7
IT_companies after adding Twitter: {'Google', 'IBM', 'Amazon', 'Twitter', 'Oracle', 'Facebook', 'Microsoft', 'Apple'}
IT_companies after adding multiple companies: {'Snapchat', 'Oracle', 'Amazon', 'Facebook', 'IBM', 'TikTok', 'Twitter', 'Google', 'Apple', 'Spotify', 'Microsoft'}
IT_companies after removing a company: {'Snapchat', 'Amazon', 'Facebook', 'IBM', 'TikTok', 'Twitter', 'Google', 'Apple', 'Spotify', 'Microsoft'}
A union B: {19, 20, 22, 24, 25, 26, 27, 28}
A intersection B: {19, 20, 22, 24, 25, 26}
Is A a subset of B: True
Are A and B disjoint: False
A union B (again): {19, 20, 22, 24, 25, 26, 27, 28}
B union A: {19, 20, 22, 24, 25, 26, 27, 28}
Symmetric difference between A and B: {27, 28}
Length of age list: 8
Length of age set: 5
```

GITHUB REPO: https://github.com/niharika0912/BDA.git

YOUTUBE URL: https://youtu.be/9JCLohnZXZM