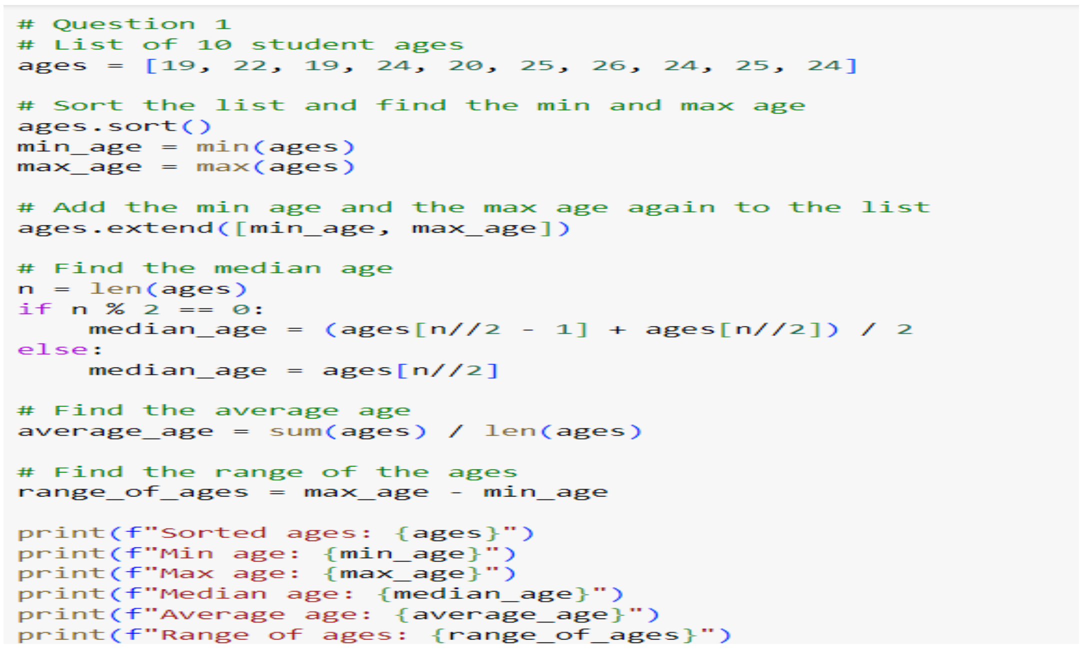
Machine Learning (ICP # 1)

Name: Preksha Reddy

700#: 700759508

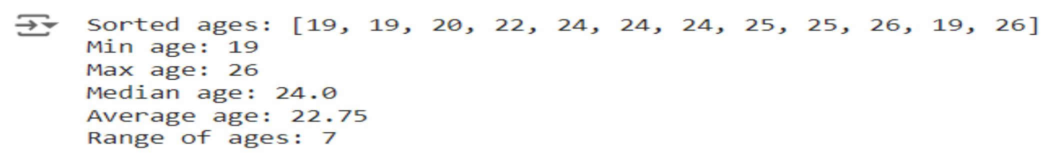
https://colab.research.google.com/drive/1L3gKD0EMyKSSojWd3HzPXbcFTuZAeoSt#scrollTo=1c1d2aZ3cwWf

Question 1:



```
# Question 1
# List of 10 student ages
ages = [19, 22, 19, 24, 20, 25, 26, 24, 25, 24]

# Sort the list and find the min and max age
ages.sort()
min_age = min(ages)
max_age = max(ages)

# Add the min age and the max age again to the list
ages.extend([min_age, max_age])

# Find the median age
n = len(ages)
if n % 2 == 0:
    median_age = (ages[n//2 - 1] + ages[n//2]) / 2
else:
    median_age = ages[n//2]

# Find the average age
average_age = sum(ages) / len(ages)

# Find the range of the ages
range_of_ages = max_age - min_age

print(f"Sorted ages: {ages}")
print(f"Min age: {min_age}")
print(f"Max age: {max_age}")
print(f"Median age: {median_age}")
print(f"Average age: {average_age}")
print(f"Range of ages: {range_of_ages}")
```

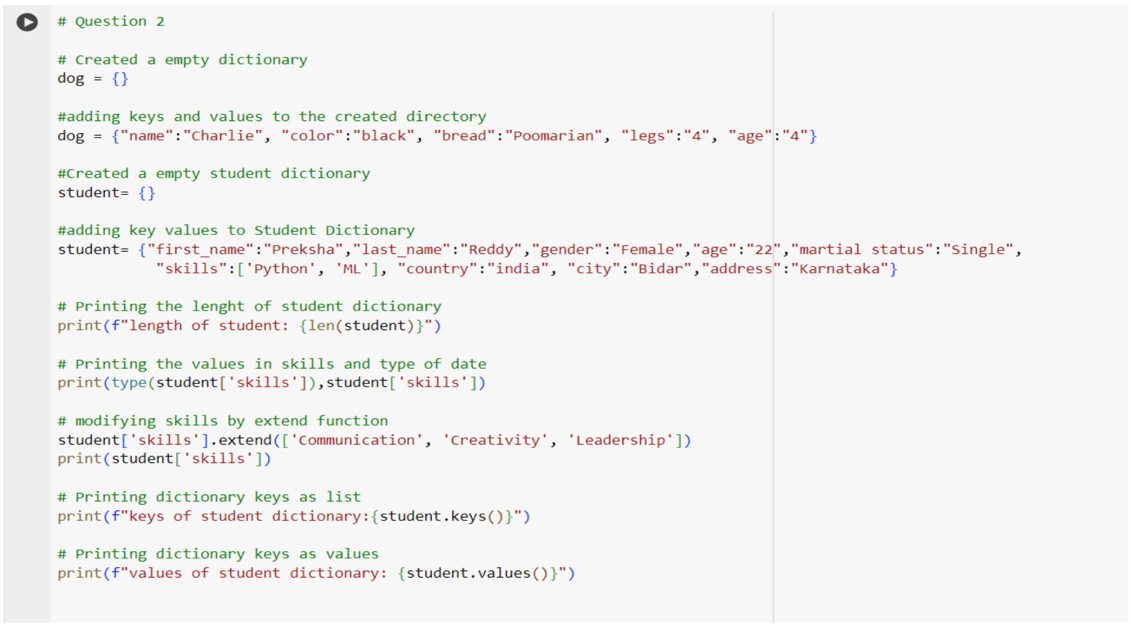
Output:



```
Sorted ages: [19, 19, 20, 22, 24, 24, 24, 25, 25, 26, 19, 26]
Min age: 19
Max age: 26
Median age: 24.0
Average age: 22.75
Range of ages: 7
```

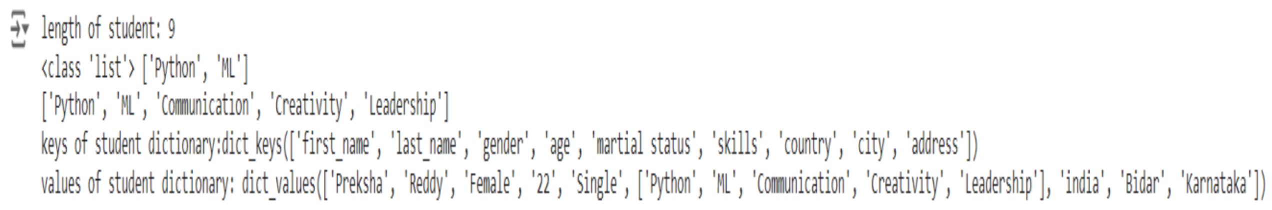
This code determines the lowest and maximum ages by first sorting the list of ages, and then adding the minimum and maximum ages back to the list. The middle value of the sorted list is found to determine the median age. The median is the average of the two middle values in a list with an even number of members. By adding together all the ages and dividing by the total number of ages, one may determine the average age. Lastly, by deducting the minimum age from the maximum age, the range of ages is determined.

Question 2:



```
# Question 2

# Created a empty dictionary
dog = {}

#adding keys and values to the created directory
dog = {"name":"Charlie", "color":"black", "bread":"Poomarian", "legs":"4", "age":"4"}

#Created a empty student dictionary
student= {}

#adding key values to Student Dictionary
student= {"first_name":"Preksha","last_name":"Reddy","gender":"Female","age":"22","martial status":"Single",
            "skills":['Python', 'ML'], "country":"india", "city":"Bidar","address":"Karnataka"}

# Printing the lenght of student dictionary
print(f"length of student: {len(student)}")

# Printing the values in skills and type of date
print(type(student['skills']),student['skills'])

# modifying skills by extend function
student['skills'].extend(['Communication', 'Creativity', 'Leadership'])
print(student['skills'])

# Printing dictionary keys as list
print(f"keys of student dictionary:{student.keys()}")

# Printing dictionary keys as values
print(f"values of student dictionary: {student.values()}")
```

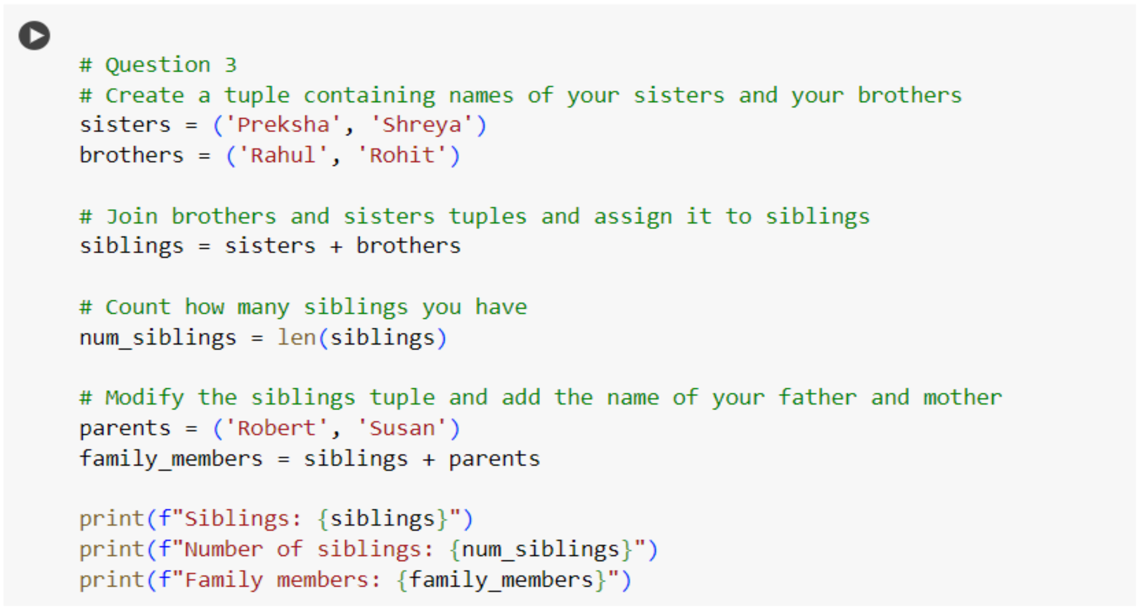
Output:



```
length of student: 9
<class 'list'> ['Python', 'ML']
['Python', 'ML', 'Communication', 'Creativity', 'Leadership']
keys of student dictionary:dict_keys(['first_name', 'last_name', 'gender', 'age', 'martial status', 'skills', 'country', 'city', 'address'])
values of student dictionary: dict_values(['Preksha', 'Reddy', 'Female', '22', 'Single', ['Python', 'ML', 'Communication', 'Creativity', 'Leadership'], 'india', 'Bidar', 'Karnataka'])
```

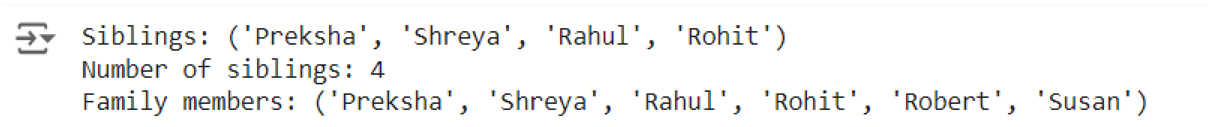
An empty dictionary dog is created and the keys and values filled with information about a dog are added. In a similar manner, empty student dictionary is generated and loaded with data on a student. The kind and contents of the skills list are printed after the length of the student dictionary. Then, more skills are added to the list of abilities. In order to show how to access and display dictionary data, the student dictionary's keys and values are written at the end.

Question 3:



```
# Question 3
# Create a tuple containing names of your sisters and your brothers
sisters = ('Preksha', 'Shreya')
brothers = ('Rahul', 'Rohit')

# Join brothers and sisters tuples and assign it to siblings
siblings = sisters + brothers

# Count how many siblings you have
num_siblings = len(siblings)

# Modify the siblings tuple and add the name of your father and mother
parents = ('Robert', 'Susan')
family_members = siblings + parents

print(f"Siblings: {siblings}")
print(f"Number of siblings: {num_siblings}")
print(f"Family members: {family_members}")
```

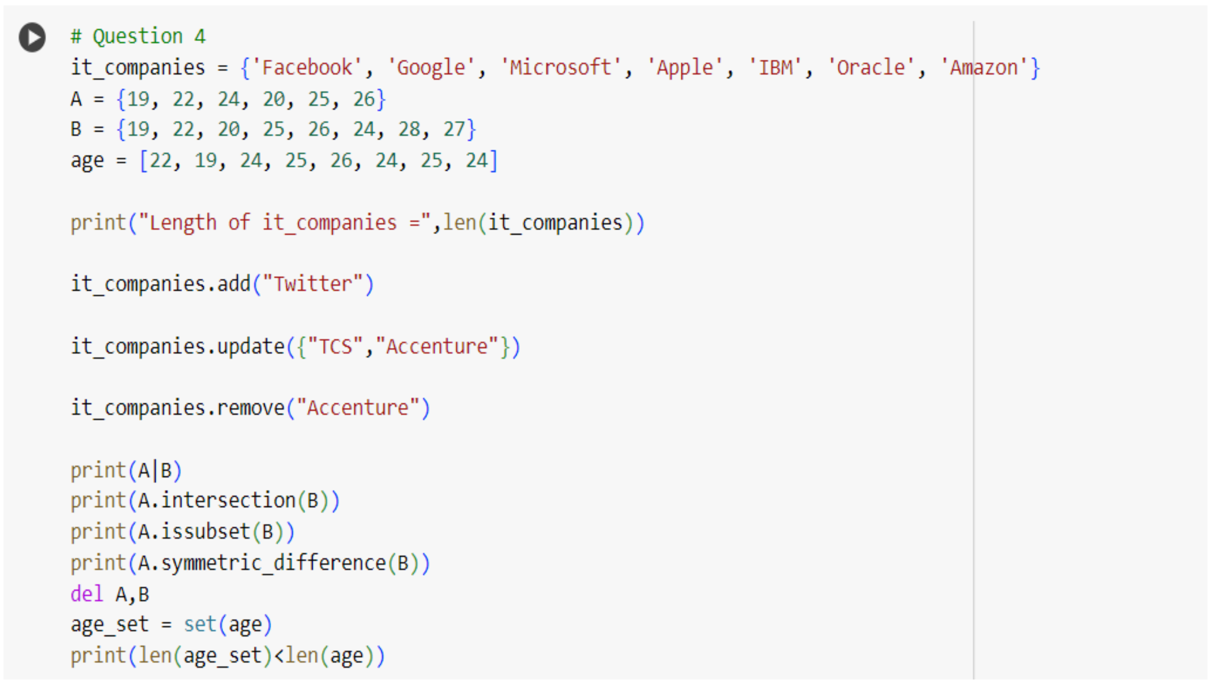
Output:



```
Siblings: ('Preksha', 'Shreya', 'Rahul', 'Rohit')
Number of siblings: 4
Family members: ('Preksha', 'Shreya', 'Rahul', 'Rohit', 'Robert', 'Susan')
```

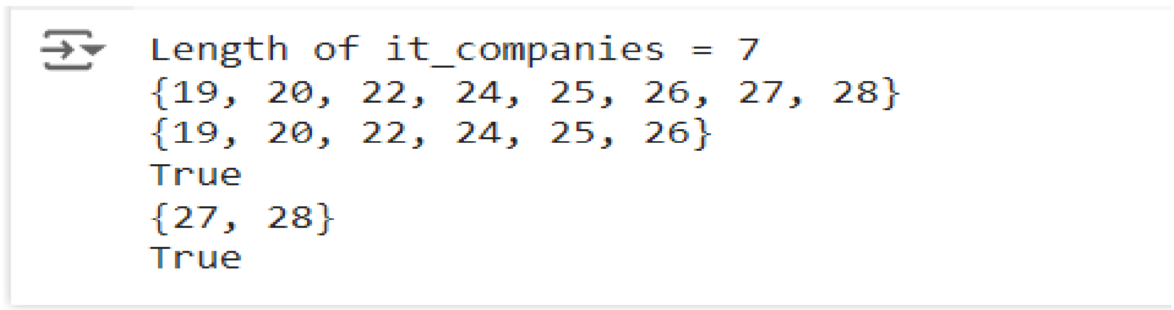
The code generates tuples for brothers and sisters, then merges them to produce a new tuple including all siblings. It determines the entire count of siblings. Next, it builds a tuple for each parent and joins it with the tuple for each sibling to generate a new tuple for family members. It produces a new tuple to include the parents because tuples are permanent. Lastly, the number of siblings and all of the family members are printed.

Question 4:



```python
# Question 4
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]

print("Length of it_companies =",len(it_companies))

it_companies.add("Twitter")

it_companies.update({"TCS","Accenture"})

it_companies.remove("Accenture")

print(A|B)
print(A.intersection(B))
print(A.issubset(B))
print(A.symmetric_difference(B))
del A,B
age_set = set(age)
print(len(age_set)<len(age))
```

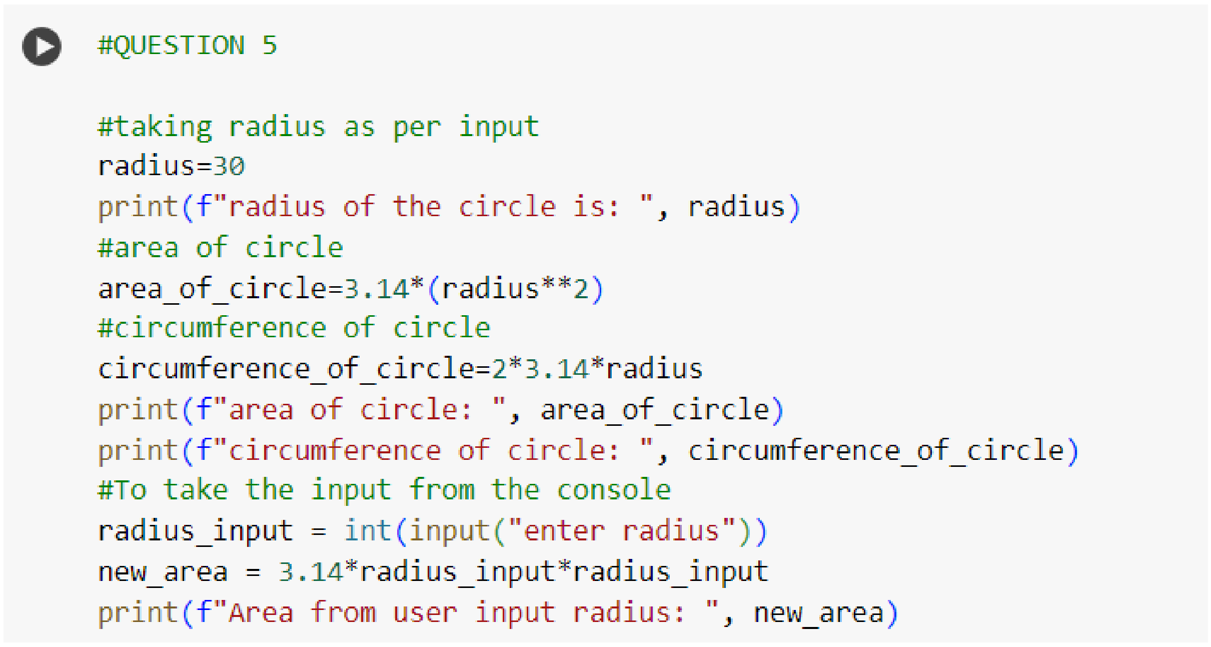
Output:



```
Length of it_companies = 7
{19, 20, 22, 24, 25, 26, 27, 28}
{19, 20, 22, 24, 25, 26}
True
{27, 28}
True
```

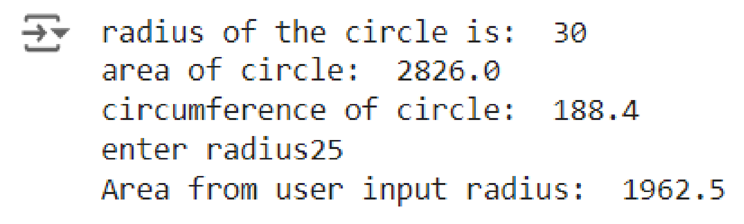
The code manages sets of companies and ages, finding lengths, adding elements, performing set operations (union, intersection, subset check, symmetric difference), removing duplicates, and checking for duplicates in ages.

Question 5:



```python
#QUESTION 5

#taking radius as per input
radius=30
print(f"radius of the circle is: ", radius)
#area of circle
area_of_circle=3.14*(radius**2)
#circumference of circle
circumference_of_circle=2*3.14*radius
print(f"area of circle: ", area_of_circle)
print(f"circumference of circle: ", circumference_of_circle)
#To take the input from the console
radius_input = int(input("enter radius"))
new_area = 3.14*radius_input*radius_input
print(f"Area from user input radius: ", new_area)
```

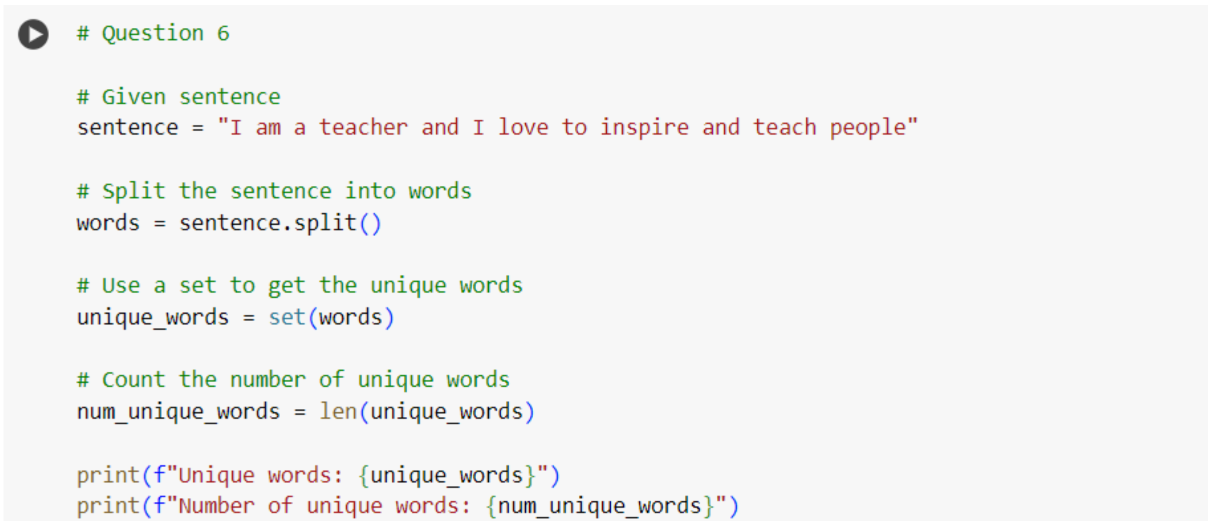
Output:



```
radius of the circle is:  30
area of circle:  2826.0
circumference of circle:  188.4
enter radius25
Area from user input radius:  1962.5
```

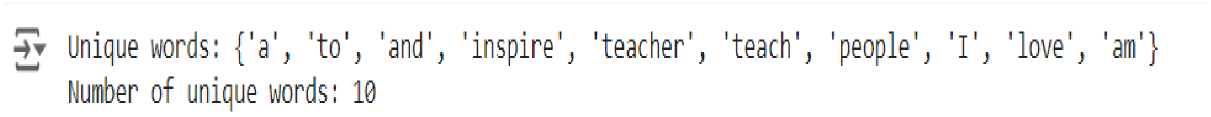
The area and circumference of a circle are calculated in this code. It determines the area and circumference first, then sets a radius value. Then, it requests that you enter a radius value, using the information you provide to compute the area.

Question 6:



```python
# Question 6

# Given sentence
sentence = "I am a teacher and I love to inspire and teach people"

# Split the sentence into words
words = sentence.split()

# Use a set to get the unique words
unique_words = set(words)

# Count the number of unique words
num_unique_words = len(unique_words)

print(f"Unique words: {unique_words}")
print(f"Number of unique words: {num_unique_words}")
```

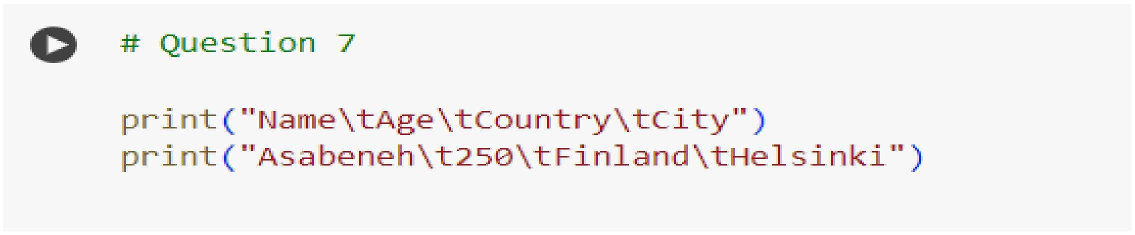
Output:



```
Unique words: {'a', 'to', 'and', 'inspire', 'teacher', 'teach', 'people', 'I', 'love', 'am'}
Number of unique words: 10
```

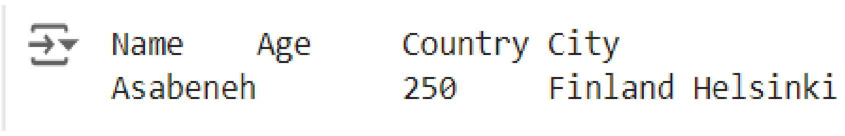
In this code each unique word in a sentence is counted. It breaks the statement up into words, then eliminates duplicates using a set. The number of unique words is then determined by counting the number of words in the set.

Question 7:



```python
# Question 7

print("Name\tAge\tCountry\tCity")
print("Asabeneh\t250\tFinland\tHelsinki")
```

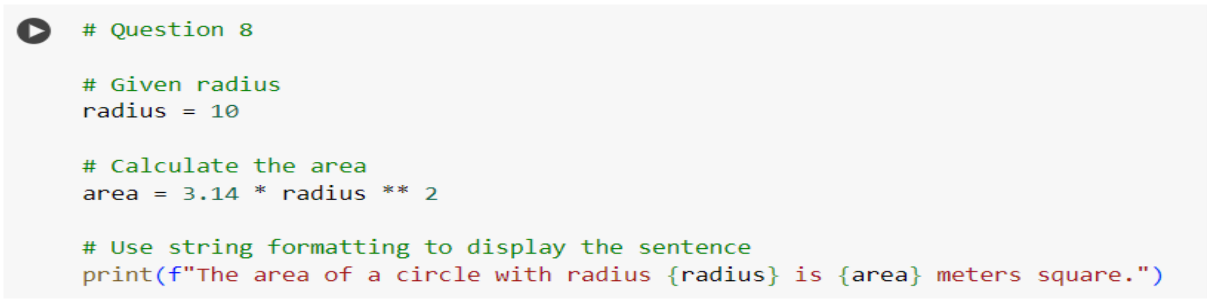
Output:



```
Name    Age     Country City
Asabeneh        250     Finland Helsinki
```

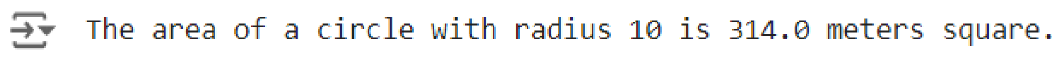
This code generates a basic table to show a person's details. Using tab characters (\t) to create space between the labels for "Name," "Age," "Country," and "City," the first line functions as a table header and the second line filled with person's actual data filled in the database.

Question 8:



```python
# Question 8

# Given radius
radius = 10

# Calculate the area
area = 3.14 * radius ** 2

# Use string formatting to display the sentence
print(f"The area of a circle with radius {radius} is {area} meters square.")
```

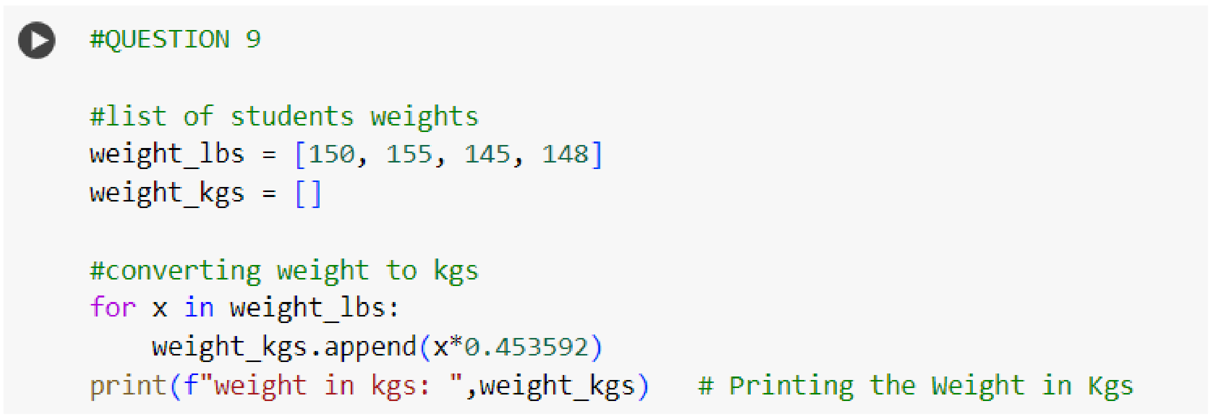
Output:



```
The area of a circle with radius 10 is 314.0 meters square.
```

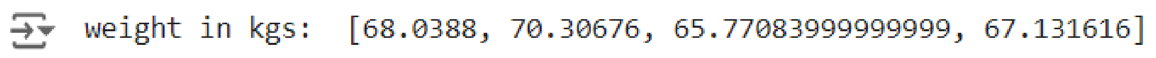
In this code the area of a circle is determined. After setting the radius to 10, the area is calculated using the formula area = 3.14 * radius**2. Lastly, it uses f-strings formatted to display the sentence.

Question 9:



```python
#QUESTION 9

#list of students weights
weight_lbs = [150, 155, 145, 148]
weight_kgs = []

#converting weight to kgs
for x in weight_lbs:
    weight_kgs.append(x*0.453592)
print(f"weight in kgs: ",weight_kgs)   # Printing the Weight in Kgs
```

Output:



```
weight in kgs:  [68.0388, 70.30676, 65.77083999999999, 67.131616]
```

A list of weights is converted from pounds (lbs) to kilograms (kgs). It features an empty list for kilograms and a list of weights in pounds. It goes over each weight in pounds, multiplies it by a certain amount to convert it to kilograms, and then adds the resultant weight to the list of kilograms. The list of weights in kilos is displayed at the end.