Lab-5

1. The following is a list of 10 students ages: ages = [19, 22, 19, 24, 20, 25, 26, 24, 25, 24]I. Sort the list and find the min and max age.
II. Add the min age and the max age again to the list.
III. Find the median age (one middle item or two middle items divided by two).IV. Find the average age (sum of all items divided by their number).
V. Find the range of the ages (max minus min).
VI. Compare the value of (min - average) and (max - average), use abs() method.
->

```
ages = [19, 22, 19, 24, 20, 25, 26, 24, 25, 24]
ages.sort()
min_age = ages[0]
max_age = ages[-1]
ages.append(min_age)
ages.append(max_age)
n = len(ages)
if n % 2 == 0:
    median_age = (ages[n//2 - 1] + ages[n//2]) / 2
else:
    median_age = ages[n//2]
average_age = sum(ages) / len(ages)
age_range = max_age - min_age
min_diff = abs(min_age - average_age)
max_diff = abs(max_age - average_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:.2f}")
print(f"Range of ages: {age_range}")
print(f"Difference between min age and average: {min_diff:.2f}")
print(f"Difference between max age and average: {max_diff:.2f}")
ages = [19, 22, 19, 24, 20, 25, 26, 24, 25, 24]
ages.sort()
min_age = ages[0]
max_age = ages[-1]
ages.append(min_age)
ages.append(max_age)
n = len(ages)
if n % 2 == 0:
    median_age = (ages[n//2 - 1] + ages[n//2]) / 2
else:
    median_age = ages[n//2]
average_age = sum(ages) / len(ages)
age_range = max_age - min_age
min_diff = abs(min_age - average_age)
max_diff = abs(max_age - average_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:.2f}")
print(f"Range of ages: {age_range}")
print(f"Difference between min age and average: {min_diff:.2f}")
print(f"Difference between max age and average: {max_diff:.2f}")
```

```
PS C:\Users\anupa\OneDrive\Desktop\My workplace> python -u "c:\Users\anupa\OneDrive\Desktop\My workplace\python\Lab5\Q_1.py"
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
Difference between min age and average: 3.75
Difference between max age and average: 3.25
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
Difference between min age and average: 3.75
Difference between max age and average: 3.25
PS C:\Users\anupa\OneDrive\Desktop\My workplace>
```

2 Iterate through the list, ['Python', 'Numpy', 'Pandas', 'Django', 'Flask'] using a for loop and print outthe items.

->

```
technologies = ['Python', 'Numpy', 'Pandas', 'Django', 'Flask']
for tech in technologies:
   print(tech)
technologies = ['Python', 'Numpy', 'Pandas', 'Django', 'Flask']
for tech in technologies:
   print(tech)
```

```
PS C:\Users\anupa\OneDrive\Desktop\My workplace> python -u "c:\Users\anupa\OneDrive\Desktop\My workplace\python\Lab5\Q_2.py"
Python
Numpy
Pandas
Django
Flask
Python
Numpy
Pandas
Django
Flask
PS C:\Users\anupa\OneDrive\Desktop\My workplace>
```

3 Create fruits, vegetables and animal products tuples.
I. Join the three tuples and assign it to a variable called food_stuff_tp.II. Change the about food_stuff_tp tuple to a food_stuff_lt list.
III. Slice out the middle item or items from the food_stuff_tp tuple or food_stuff_lt list.IV. Slice out the first three items and the last three items from food_staff_lt list.
V. Delete the food_staff_tp tuple completely.

->

```
fruits = ('apple', 'banana', 'orange')
vegetables = ('carrot', 'broccoli', 'spinach')
animal_products = ('milk', 'eggs', 'cheese')
food_stuff_tp = fruits + vegetables + animal_products
food_stuff_lt = list(food_stuff_tp)
n = len(food_stuff_lt)
if n % 2 == 0:
   middle_items = food_stuff_lt[n//2 - 1:n//2 + 1]
else:
   middle_items = food_stuff_lt[n//2]
first_three_items = food_stuff_lt[:3]
last_three_items = food_stuff_lt[-3:]
del food_stuff_tp
print(f"Combined food_stuff_tp tuple: {food_stuff_lt}")
print(f"Middle item(s): {middle_items}")
print(f"First three items: {first_three_items}")
print(f"Last three items: {last_three_items}")
fruits = ('apple', 'banana', 'orange')
vegetables = ('carrot', 'broccoli', 'spinach')
animal_products = ('milk', 'eggs', 'cheese')
food_stuff_tp = fruits + vegetables + animal_products
food_stuff_lt = list(food_stuff_tp)
n = len(food_stuff_lt)
if n % 2 == 0:
   middle_items = food_stuff_lt[n//2 - 1:n//2 + 1]
else:
   middle_items = food_stuff_lt[n//2]
first_three_items = food_stuff_lt[:3]
last_three_items = food_stuff_lt[-3:]
del food_stuff_tp
print(f"Combined food_stuff_tp tuple: {food_stuff_lt}")
print(f"Middle item(s): {middle_items}")
print(f"First three items: {first_three_items}")
print(f"Last three items: {last_three_items}")
```

```
PS C:\Users\anupa\OneDrive\Desktop\My workplace> python -u "c:\Users\anupa\OneDrive\Desktop\My workplace\python\Lab5\Q_3.py"
Combined food_stuff_tp tuple: ['apple', 'banana', 'orange', 'carrot', 'broccoli', 'spinach', 'milk', 'eggs', 'cheese']
Middle item(s): broccoli
First three items: ['apple', 'banana', 'orange']
Last three items: ['milk', 'eggs', 'cheese']
Combined food_stuff_tp tuple: ['apple', 'banana', 'orange', 'carrot', 'broccoli', 'spinach', 'milk', 'eggs', 'cheese']
Middle item(s): broccoli
First three items: ['apple', 'banana', 'orange']
Last three items: ['milk', 'eggs', 'cheese']
PS C:\Users\anupa\OneDrive\Desktop\My workplace>
```

4 Create a set given below
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]
I. Find the length of the set it_companies.
II. Add 'Twitter' to it_companies.
III. Insert multiple IT companies at once to the set it_companies.IV. Remove one of the companies from the set it_companies.
V. What is the difference between remove and discard.

->

```
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]
length_it_companies = len(it_companies)
it_companies.add('Twitter')
it_companies.update(['Snapchat', 'TikTok', 'LinkedIn'])
it_companies.remove('Oracle')
print(f"Length of it_companies set: {length_it_companies}")
print(f"it_companies after adding 'Twitter': {it_companies}")
try:
    it_companies.remove('Oracle')
except KeyError:
    print("'Oracle' was already removed using remove(), trying discard now...")
it_companies.discard('Oracle')
print(f"it_companies after removing 'Oracle' again: {it_companies}")

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]
length_it_companies = len(it_companies)
it_companies.add('Twitter')
it_companies.update(['Snapchat', 'TikTok', 'LinkedIn'])
it_companies.remove('Oracle')
print(f"Length of it_companies set: {length_it_companies}")
print(f"it_companies after adding 'Twitter': {it_companies}")
try:
    it_companies.remove('Oracle')
except KeyError:
    print("'Oracle' was already removed using remove(), trying discard now...")
it_companies.discard('Oracle')
print(f"it_companies after removing 'Oracle' again: {it_companies}")
```

```
PS C:\Users\anupa\OneDrive\Desktop\My workplace> python -u "c:\Users\anupa\OneDrive\Desktop\My workplace\python\Lab5\Q_4.py"
Length of it_companies set: 7
it_companies after adding 'Twitter': {'Twitter', 'Microsoft', 'Facebook', 'Snapchat', 'LinkedIn', 'TikTok', 'Google', 'Apple', 'Amazon', 'IBM'}
'Oracle' was already removed using remove(), trying discard now...
it_companies after removing 'Oracle' again: {'Twitter', 'Microsoft', 'Facebook', 'Snapchat', 'LinkedIn', 'TikTok', 'Google', 'Apple', 'Amazon', 'IBM'}
Length of it_companies set: 7
it_companies after adding 'Twitter': {'Twitter', 'Microsoft', 'Facebook', 'Snapchat', 'LinkedIn', 'TikTok', 'Google', 'Apple', 'Amazon', 'IBM'}
'Oracle' was already removed using remove(), trying discard now...
it_companies after removing 'Oracle' again: {'Twitter', 'Microsoft', 'Facebook', 'Snapchat', 'LinkedIn', 'TikTok', 'Google', 'Apple', 'Amazon', 'IBM'}
```

5 From the above sets A and BI. Join A and B
II. Find A intersection BIII. Is A subset of B
IV. Are A and B disjoint sets V. Join A with B and B with A
VI. What is the symmetric difference between A and BVII. Delete the sets completely

->

```
A = {19, 22, 24, 20, 25, 26}
B = {19, 22, 20, 25, 26, 24, 28, 27}
A_union_B = A.union(B)
A_intersection_B = A.intersection(B)
is_A_subset_of_B = A.issubset(B)
```

```
are_A_and_B_disjoint = A.isdisjoint(B)
A_joined_with_B = A.union(B)
B_joined_with_A = B.union(A)
symmetric_difference_A_B = A.symmetric_difference(B)
del A
del B
print(f"Union of A and B: {A_union_B}")
print(f"Intersection of A and B: {A_intersection_B}")
print(f"Is A a subset of B? {is_A_subset_of_B}")
print(f"Are A and B disjoint sets? {are_A_and_B_disjoint}")
print(f"Join A with B: {A_joined_with_B}")
print(f"Join B with A: {B_joined_with_A}")
print(f"Symmetric difference between A and B: {symmetric_difference_A_B}")
try:
    print(A)
    print(B)
except NameError:
    print("Sets A and B have been deleted and no longer exist.")


A = {19, 22, 24, 20, 25, 26}
B = {19, 22, 20, 25, 26, 24, 28, 27}
A_union_B = A.union(B)
A_intersection_B = A.intersection(B)
is_A_subset_of_B = A.issubset(B)
are_A_and_B_disjoint = A.isdisjoint(B)
A_joined_with_B = A.union(B)
B_joined_with_A = B.union(A)
symmetric_difference_A_B = A.symmetric_difference(B)
del A
del B
print(f"Union of A and B: {A_union_B}")
print(f"Intersection of A and B: {A_intersection_B}")
print(f"Is A a subset of B? {is_A_subset_of_B}")
print(f"Are A and B disjoint sets? {are_A_and_B_disjoint}")
print(f"Join A with B: {A_joined_with_B}")
print(f"Join B with A: {B_joined_with_A}")
print(f"Symmetric difference between A and B: {symmetric_difference_A_B}")
try:
    print(A)
    print(B)
except NameError:
    print("Sets A and B have been deleted and no longer exist.")
```

```
PS C:\Users\anupa\OneDrive\Desktop\My workplace> python -u "c:\Users\anupa\OneDrive\Desktop\My workplace\python\Lab5\Q_5.py"
Union of A and B: {19, 20, 22, 24, 25, 26, 27, 28}
Intersection of A and B: {19, 20, 22, 24, 25, 26}
Is A a subset of B? True
Are A and B disjoint sets? False
Join A with B: {19, 20, 22, 24, 25, 26, 27, 28}
Join B with A: {19, 20, 22, 24, 25, 26, 27, 28}
Symmetric difference between A and B: {27, 28}
Sets A and B have been deleted and no longer exist.
Union of A and B: {19, 20, 22, 24, 25, 26, 27, 28}
Intersection of A and B: {19, 20, 22, 24, 25, 26}
Is A a subset of B? True
Are A and B disjoint sets? False
Join A with B: {19, 20, 22, 24, 25, 26, 27, 28}
Join B with A: {19, 20, 22, 24, 25, 26, 27, 28}
Symmetric difference between A and B: {27, 28}
Sets A and B have been deleted and no longer exist.
```

6. Create an empty dictionary called dog.Add name, color, breed, legs, age to the dog dictionary.

->

```
dog = {}
dog['name'] = 'Buddy'
dog['color'] = 'Brown'
dog['breed'] = 'Golden Retriever'
dog['legs'] = 4
dog['age'] = 3
print(dog)
dog = {}
dog['name'] = 'Buddy'
dog['color'] = 'Brown'
dog['breed'] = 'Golden Retriever'
dog['legs'] = 4
dog['age'] = 3
```

```
PS C:\Users\anupa\OneDrive\Desktop\My workplace> python -u "c:\Users\anupa\OneDrive\Desktop\My workplace\python\Lab5\Q_6.py"
{'name': 'Buddy', 'color': 'Brown', 'breed': 'Golden Retriever', 'legs': 4, 'age': 3}
PS C:\Users\anupa\OneDrive\Desktop\My workplace>
```

7. Create a student dictionary and add first_name, last_name, gender, age, marital status, skills, country, city and address as keys for the dictionary
I. Get the length of the student dictionary.
II. Get the value of skills and check the data type, it should be a list.III. Modify the skills values by adding one or two skills.
IV. Get the dictionary keys as a list. V. Get the dictionary values as a list.
VI. Change the dictionary to a list of tuples using items() method.VII. Delete one of the items in the dictionary.
VIII. Delete one of the dictionaries.

->

```
student = {
    'first_name': 'John',
    'last_name': 'Doe',
    'gender': 'Male',
    'age': 21,
    'marital_status': 'Single',
    'skills': ['Python', 'Data Analysis'],
    'country': 'USA',
    'city': 'New York',
    'address': '123 Main St'
}
length_of_student_dict = len(student)
skills = student['skills']
skills_data_type = type(skills)
student['skills'].extend(['Machine Learning', 'Web Development'])
keys_list = list(student.keys())
values_list = list(student.values())
student_items = list(student.items())
del student['address']
del student
print(f"Length of student dictionary: {length_of_student_dict}")
print(f"Skills: {skills}")
print(f"Data type of skills: {skills_data_type}")
print(f"Modified skills: {skills}")
print(f"Keys in student dictionary: {keys_list}")
print(f"Values in student dictionary: {values_list}")
print(f"Student dictionary as list of tuples: {student_items}")
try:
    print(student)
except NameError:
    print("The student dictionary has been deleted and no longer exists.")

student = {
    'first_name': 'John',
    'last_name': 'Doe',
    'gender': 'Male',
    'age': 21,
    'marital_status': 'Single',
    'skills': ['Python', 'Data Analysis'],
    'country': 'USA',
    'city': 'New York',
    'address': '123 Main St'
}
length_of_student_dict = len(student)
skills = student['skills']
skills_data_type = type(skills)
student['skills'].extend(['Machine Learning', 'Web Development'])
keys_list = list(student.keys())
values_list = list(student.values())
student_items = list(student.items())
del student['address']
del student
```

```
print(f"Length of student dictionary: {length_of_student_dict}")
print(f"Skills: {skills}")
print(f"Data type of skills: {skills_data_type}")
print(f"Modified skills: {skills}")
print(f"Keys in student dictionary: {keys_list}")
print(f"Values in student dictionary: {values_list}")
print(f"Student dictionary as list of tuples: {student_items}")
try:
    print(student)
except NameError:
    print("The student dictionary has been deleted and no longer exists.")
```

```
PS C:\Users\anupa\OneDrive\Desktop\My workplace> python -u "c:\Users\anupa\OneDrive\Desktop\My workplace\python\Lab5\Q_7.py"
Length of student dictionary: 9
Skills: ['Python', 'Data Analysis', 'Machine Learning', 'Web Development']
Data type of skills: <class 'list'>
Modified skills: ['Python', 'Data Analysis', 'Machine Learning', 'Web Development']
Keys in student dictionary: ['first_name', 'last_name', 'gender', 'age', 'marital_status', 'skills', 'country', 'city', 'address']
Values in student dictionary: ['John', 'Doe', 'Male', 21, 'Single', ['Python', 'Data Analysis', 'Machine Learning', 'Web Development'], 'USA', 'New York', '123 Main St']
Student dictionary as list of tuples: [('first_name', 'John'), ('last_name', 'Doe'), ('gender', 'Male'), ('age', 21), ('marital_status', 'Single'), ('skills', ['Python', 'Data Analysis', 'Machine Lea
rning', 'Web Development']), ('country', 'USA'), ('city', 'New York'), ('address', '123 Main St')]
The student dictionary has been deleted and no longer exists.
Length of student dictionary: 9
Skills: ['Python', 'Data Analysis', 'Machine Learning', 'Web Development']
Data type of skills: <class 'list'>
Modified skills: ['Python', 'Data Analysis', 'Machine Learning', 'Web Development']
Keys in student dictionary: ['first_name', 'last_name', 'gender', 'age', 'marital_status', 'skills', 'country', 'city', 'address']
Values in student dictionary: ['John', 'Doe', 'Male', 21, 'Single', ['Python', 'Data Analysis', 'Machine Learning', 'Web Development'], 'USA', 'New York', '123 Main St']
Student dictionary as list of tuples: [('first_name', 'John'), ('last_name', 'Doe'), ('gender', 'Male'), ('age', 21), ('marital_status', 'Single'), ('skills', ['Python', 'Data Analysis', 'Machine Lea
rning', 'Web Development']), ('country', 'USA'), ('city', 'New York'), ('address', '123 Main St')]
The student dictionary has been deleted and no longer exists.
```

8. Create a person dictionary. person={ 'first_name': 'Asabeneh', 'last_name': 'Yetayeh', 'age': 250, 'country': 'Finland', 'is_marred': True, 'skills': ['JavaScript', 'React', 'Node', 'Mongo', 'Python'], 'address:' { 'street': 'Space street', 'zipcode':'02210';}}

I. Check if the person dictionary has skills key, if so print out the middle skill in the skills list.

II. Check if the person dictionary has skills key, if so check if the person has 'Python' skill and print out the result.

III. If a person skills has only JavaScript and React, print('He is a front end developer'), if the person skills has Node, Python, MongoDB, print('He is a backend developer'), if the person skills has React, Node and MongoDB, Print('He is a fullstack developer'), else print('unknowntitle') - for more accurate results more conditions can be nested!

IV. If the person is married and if he lives in Finland, print the information in the following format:

```py
Asabeneh Yetayeh lives in Finland. He is married.
```

->

```
person = {
    'first_name': 'Asabeneh',
    'last_name': 'Yetayeh',
    'age': 250,
    'country': 'Finland',
    'is_marred': True,
    'skills': ['JavaScript', 'React', 'Node', 'MongoDB', 'Python'],
    'address': {
        'street': 'Space street',
        'zipcode': '02210'
    }}
if 'skills' in person:
    skills = person['skills']
    middle_index = len(skills) // 2
    middle_skill = skills[middle_index]
    print(f"Middle skill: {middle_skill}")
if 'skills' in person:
    has_python = 'Python' in person['skills']
    print(f"Has Python skill: {has_python}")
if 'skills' in person:
    skills = person['skills']
    if set(['JavaScript', 'React']).issubset(skills) and len(skills) == 2:
        print("He is a front end developer")
    elif set(['Node', 'Python', 'MongoDB']).issubset(skills):
        print("He is a backend developer")
    elif set(['React', 'Node', 'MongoDB']).issubset(skills):
        print("He is a fullstack developer")
    else:
        print("Unknown title")
if person['is_marred'] and person['country'] == 'Finland':
```

```
    print(f"{person['first_name']} {person['last_name']} lives in Finland. He is married.")
person = {
    'first_name': 'Asabeneh',
    'last_name': 'Yetayeh',
    'age': 250,
    'country': 'Finland',
    'is_marred': True,
    'skills': ['JavaScript', 'React', 'Node', 'MongoDB', 'Python'],
    'address': {
        'street': 'Space street',
        'zipcode': '02210'
    }}
if 'skills' in person:
    skills = person['skills']
    middle_index = len(skills) // 2
    middle_skill = skills[middle_index]
    print(f"Middle skill: {middle_skill}")
if 'skills' in person:
    has_python = 'Python' in person['skills']
    print(f"Has Python skill: {has_python}")
if 'skills' in person:
    skills = person['skills']
    if set(['JavaScript', 'React']).issubset(skills) and len(skills) == 2:
        print("He is a front end developer")
    elif set(['Node', 'Python', 'MongoDB']).issubset(skills):
        print("He is a backend developer")
    elif set(['React', 'Node', 'MongoDB']).issubset(skills):
        print("He is a fullstack developer")
    else:
        print("Unknown title")
if person['is_marred'] and person['country'] == 'Finland':
    print(f"{person['first_name']} {person['last_name']} lives in Finland. He is married.")
```

```
PS C:\Users\anupa\OneDrive\Desktop\My workplace> python -u "c:\Users\anupa\OneDrive\Desktop\My workplace\python\Lab5\Q_8.py
Middle skill: Node
Has Python skill: True
He is a backend developer
Asabeneh Yetayeh lives in Finland. He is married.
Middle skill: Node
Has Python skill: True
He is a backend developer
Asabeneh Yetayeh lives in Finland. He is married.
```

9. Print the season name of the year based on the month number using a dictionary.

->

```
month_to_season = {
    1: 'Winter', 2: 'Winter', 3: 'Spring',
    4: 'Spring', 5: 'Spring', 6: 'Summer',
    7: 'Summer', 8: 'Summer', 9: 'Fall',
    10: 'Fall', 11: 'Fall', 12: 'Winter'
}
def get_season(month):
    return month_to_season.get(month, 'Invalid month number')
month_number = 7
season = get_season(month_number)
print(f"Month {month_number} is in the {season} season.")
month_to_season = {
    1: 'Winter', 2: 'Winter', 3: 'Spring',
    4: 'Spring', 5: 'Spring', 6: 'Summer',
    7: 'Summer', 8: 'Summer', 9: 'Fall',
    10: 'Fall', 11: 'Fall', 12: 'Winter'
}
def get_season(month):
    return month_to_season.get(month, 'Invalid month number')
month_number = 7
season = get_season(month_number)
print(f"Month {month_number} is in the {season} season.")
```

```
PS C:\Users\anupa\OneDrive\Desktop\My workplace> python -u "c:\Users\anupa\OneDrive\Desktop\My workplace\python\Lab5\Q_9.py"
Month 7 is in the Summer season.
Month 7 is in the Summer season.
PS C:\Users\anupa\OneDrive\Desktop\My workplace>
```