Sample Coding Questions

Dictionaries

Dictionaries are used to store data values in key:value pairs.

A dictionary is a collection which is ordered*, changeable and do not allow duplicates.

Dictionaries are written with curly brackets, and have keys and values:

```
MyDict =      {
  "university": "GLA University",
  "location": "Mathura",
  "year": 2010
}
print(MyDict)
print(MyDict["year"])
print(len(MyDict))
print(type(MyDict))


thisdict = dict(name = "John", age = 36, country = "Norway")
print(thisdict)
```

```
MyDict =      {
  "university": "GLA University",
  "location": "Mathura",
  "year": 2010
}
#print(MyDict)
check = MyDict["year"]
print(check)

x = MyDict.get("year")
print(x)



x1 = MyDict.keys()
print(x1)



x2 = MyDict.values()
print(x2)



car = {
"brand": "Ford",
"model": "Mustang",
"year": 1964
}
```

```
x = car.keys()    #car.values()

print(x) #before the change

car["color"] = "white"
print(car)
print(x) #after the change



thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}

x = thisdict.items()
print(thisdict)
print(x)


thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
```

```
}

if "model" in thisdict:
  print("Yes, 'model' is one of the keys in the thisdict dictionary")



if "Mustang" in thisdict.values():
  print("Yes, 'Mustang' is one of the values in the thisdict dictionary")




students = ('Sachin', 'Ram', 'Laxman')
marks = 0

StudDict = dict.fromkeys(students, marks )

print(StudDict)




thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
```

```
thisdict["color"] = "red"
print(thisdict)
```

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
thisdict.update({"color": "red"})
```

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}

for x in thisdict:
  print(x)
```

```
for x in thisdict:
  print(thisdict[x])
```

```
for x in thisdict.values():
  print(x)


for x in thisdict.keys():
  print(x)


for x, y in thisdict.items():
  print(x, y)



thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
thisdict.pop("model")
print(thisdict)


thisdict = {
  "brand": "Ford",
  "model": "Mustang",
```

```
  "year": 1964
}
thisdict.popitem()
print("My Data",thisdict)



thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
del thisdict["model"]
print(thisdict)



thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
del thisdict
print(thisdict) #this will cause an error because "thisdict" no longer exists.



thisdict = {
  "brand": "Ford",
```

```
  "model": "Mustang",
  "year": 1964
}
thisdict.clear()
print(thisdict)



MyDetails = {
 "university": "GLA University",
  "location": "Mathura",
  "year": 2010
}
MyDetails["year"] = 2018

print(MyDetails)



thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
thisdict.update({"year": 2020})
print(thisdict)
```

```
myStudents = {
  "stud1" : {
    "name" : "Sachin",
    "year" : 2004
  },
  "stud2" : {
    "name" : "Laxman",
    "year" : 2007
  },
  "stud3" : {
    "name" : "Ram",
    "year" : 2011
  }
}

print(myStudents)

for x in myStudents:
    print(x)

print(myStudents["stud2"]["name"])

for x, y in myStudents.items():
    print(x, y)
```

Problem Statement:

You are required to create a Python program that allows a user to input key-value pairs to construct a

#dictionary. The program should follow these steps:

1. Prompt the user to enter the number of key-value pairs they want to add.

2. For each pair, prompt the user to enter a key and its corresponding value.

3. Construct a dictionary with the provided key-value pairs.

4. Print out the resulting dictionary.

Your program should be able to handle any number of key-value pairs specified by the user.

```
# Initialize an empty dictionary
user_dict = {}

# Get user input for key-value pairs
num_pairs = int(input("Enter the number of key-value pairs you want to add: "))

for x in range(num_pairs):
    key = input("Enter the key: ")
    value = input("Enter the value: ")
    user_dict[key] = value   #user_dict.update({key:value})
```

```
# Print the resulting dictionary
print("Resulting dictionary:")
print(user_dict)



# Assignment 1: Creating and Manipulating Dictionaries
# Problem Statement:
# Create a dictionary named student_scores with the following keys and values:

# "John": 85
# "Jane": 90
# "Bob": 75
# "Alice": 95
# Perform the following operations:

# Add a new student, "Sam", with a score of 80.
# Update Bob's score to 80.
# Remove Jane from the dictionary.
# Print the names of all students and their scores.



# Creating the dictionary
student_scores = {
    "John": 85,
    "Jane": 90,
```

```
    "Bob": 75,
    "Alice": 95
}

# Adding a new student
student_scores["Sam"] = 80

# Updating Bob's score
student_scores["Bob"] = 80

# Removing Jane
del student_scores["Jane"]

# Printing names and scores
for name, score in student_scores.items():
    print(f"{name}: {score}")




# Assignment 2: Dictionary Operations
# Problem Statement:
# Given the dictionary inventory representing items in a store, perform the following operations:

# inventory = {
#     "apple": 10,
#     "banana": 5,
```

```
#     "orange": 8,
#     "grape": 3
# }

# 1. Add 2 more apples to the inventory.
# 2. Check if "banana" is in the inventory. Print a message accordingly.
# 3. Reduce the quantity of grapes by 1.
# 4. Print the total number of items in the inventory.

# Given dictionary
inventory = {
    "apple": 10,
    "banana": 5,
    "orange": 8,
    "grape": 3
}

# Adding 2 more apples
inventory["apple"] += 2

# Checking if "banana" is in the inventory
if "banana" in inventory:
    print("Yes, banana is in the inventory.")
else:
    print("No, banana is not in the inventory.")
```

```
# Reducing the quantity of grapes by 1
inventory["grape"] -= 1

# Printing total number of items
total_items = sum(inventory.values())
print(f"Total number of items in the inventory: {total_items}")



# Assignment 3: Dictionary Iteration and Manipulation
# Problem Statement:
# Given the dictionary fruits:

# fruits = {
#     "apple": 2,
#     "banana": 5,
#     "orange": 3,
#     "grape": 7
# }

# 1. Print the names of fruits with quantities greater than 4.
# 2. Double the quantity of oranges.
# 3. Create a new dictionary more_fruits with the following items: {"kiwi": 4, "pear": 6}.
# 4. Add the items from more_fruits to fruits.
# 5. Print the updated fruits dictionary.

# Given dictionary
```

```
fruits = {
    "apple": 2,
    "banana": 5,
    "orange": 3,
    "grape": 7
}

# Printing fruits with quantities > 4
print("Fruits with quantities > 4:")
for fruit, quantity in fruits.items():
    if quantity > 4:
        print(f"{fruit}: {quantity}")

# Doubling the quantity of oranges
fruits["orange"] *= 2

# Creating a new dictionary more_fruits
more_fruits = {
    "kiwi": 4,
    "pear": 6
}

# Adding items from more_fruits to fruits
fruits.update(more_fruits)

# Printing updated fruits dictionary
```

```
print("\nUpdated Fruits:")
for fruit, quantity in fruits.items():
    print(f"{fruit}: {quantity}")



# Assignment 4:


# Problem Statement:
# You are given a list of students and their corresponding grades.
# Create a dictionary where the keys are the student names and the values are their grades.
# Then, find and print the highest grade without using the max() function.

# Creating the dictionary
grades = {'John': 85, 'Jane': 92, 'Tom': 78, 'Emily': 95, 'Sam': 88}

# Initializing a variable to hold the highest grade
highest_grade = -1

# Iterating through the grades to find the highest
for grade in grades.values():
    if grade > highest_grade:
        highest_grade = grade

# Finding the student(s) with the highest grade
top_students = [student for student, grade in grades.items() if grade == highest_grade]
```

```
# Printing the highest grade and the student(s)
print(f"The highest grade is: {highest_grade}")
print(f"The student(s) with the highest grade: {', '.join(top_students)}")

# Assignment 5:

# Problem Statement:
# Given a list of prices of different items, create a user-defined dictionary where the keys are the item names and
# the values are their prices.
# Then, calculate and print the total cost of all the items without using the sum() function.

# Creating the dictionary
# item_prices = {'Apple': 2.5, 'Banana': 1.8, 'Orange': 3.0, 'Grapes': 4.2, 'Mango': 5.5}

item_prices = {}

# Get user input for key-value pairs
num_pairs = int(input("Enter the number of key-value pairs you want to add: "))

for x in range(num_pairs):
    key = input("Enter the key: ")
    value = int(input("Enter the value: "))
```

```
    item_prices[key] = value

# Print the resulting dictionary
print("Resulting dictionary:")
print(item_prices)

# Initializing a variable to hold the total cost
total_cost = 0

# Iterating through the prices to calculate the total cost
for price in item_prices.values():
    total_cost += price

# Printing the total cost
print(f"The total cost of all items is: {total_cost}")

# Assignment 6:

# Problem Statement:
# You have been given two lists, one containing the names of students and
# the other containing their corresponding scores in a test.
# Create a dictionary where the keys are the student names and the values are their scores.
# Then, find and print the average score.
#You are allowed to use only - LEN() function.

# Given data
```

```
students = ['John', 'Jane', 'Tom', 'Emily', 'Sam']
scores = [85, 92, 78, 95, 88]

# Creating the dictionary manually
score_dict = {}

for i in range(len(students)):
    score_dict[students[i]] = scores[i]

# Calculating the total score and counting the number of scores
total_score = 0
num_scores = 0

for score in scores:
    total_score += score
    num_scores += 1

# Calculating the average score
average_score = total_score / num_scores

# Printing the average score
print(f"The average score is: {average_score}")
```

Assignment 7:

Problem Statement:

You are given a list of books and their corresponding authors.

Create a dictionary where the keys are the book titles and the values are their authors.

Then, find and print the author of a specific book, entered by the user.

If found show the author of the book searched by the user else show not available in the dictionary.

```
# Given data
books = ['Book1', 'Book2', 'Book3', 'Book4', 'Book5']
authors = ['Author1', 'Author2', 'Author3', 'Author4', 'Author5']

book_dict = {}

for i in range(len(books)):
    book_dict[books[i]] = authors[i]

print(book_dict)

# Asking the user for a book to search
search_book = input("Enter the name of the book you want to search for: ")

# Checking if the book is in the dictionary
if search_book in book_dict:
    author_of_specific_book = book_dict[search_book]
```

```
    print(f"The author of {search_book} is: {author_of_specific_book}")
else:
    print(f"The book '{search_book}' is not found in the dictionary.")
```

Assignment 8:

Problem Statement:

You have a list of students with their corresponding grades and a list of subjects.

Write a program to create a report card for each student,

with grades for each subject and the overall average.

Sample Input:

```
# students = ['John', 'Jane', 'Tom', 'Emily', 'Sam']
# grades = {'John': {'Math': 85, 'Science': 90, 'History': 75},
#           'Jane': {'Math': 92, 'Science': 88, 'History': 85},
#           'Tom': {'Math': 78, 'Science': 82, 'History': 80},
#           'Emily': {'Math': 95, 'Science': 88, 'History': 92},
#           'Sam': {'Math': 88, 'Science': 90, 'History': 85}}
# subjects = ['Math', 'Science', 'History']
```

Sample Output:

Report card for John:

Math: 85

Science: 90

History: 75

Average Grade: 83.33333333333333

Report card for Jane:

Math: 92

Science: 88

History: 85

Average Grade: 88.33333333333333

Report card for Tom:

Math: 78

Science: 82

History: 80

Average Grade: 80.0

Report card for Emily:

Math: 95

Science: 88

History: 92

Average Grade: 91.66666666666667

Report card for Sam:

Math: 88

```
# Science: 90
# History: 85
# Average Grade: 87.66666666666667

# List of students, grades, and subjects
students = ['John', 'Jane', 'Tom', 'Emily', 'Sam']
grades = {'John': {'Math': 85, 'Science': 90, 'History': 75},
        'Jane': {'Math': 92, 'Science': 88, 'History': 85},
        'Tom': {'Math': 78, 'Science': 82, 'History': 80},
        'Emily': {'Math': 95, 'Science': 88, 'History': 92},
        'Sam': {'Math': 88, 'Science': 90, 'History': 85}}
subjects = ['Math', 'Science', 'History']



report_cards = {}

for student in students:
    grades_dict = grades[student]
    # retrieves the dictionary of subjects and grades
    # for that student from the grades dictionary.
    total_grades = sum(grades_dict.values())
    average_grade = total_grades / len(subjects)
    report_cards[student] = {'Grades': grades_dict, 'Average': average_grade}
    #adds a report card entry for the student in the report_cards dictionary.
    #It includes the grades dictionary and the average grade.
```

```
# print(report_cards, "\n")

for x,y in report_cards.items():
    print(x,"->",y)



print()



# Printing the report cards
for student, report_card in report_cards.items():
#starts a loop that iterates over each student and
# their corresponding report card in the report_cards dictionary.
    print(f"Report card for {student}:")
    for subject, grade in report_card['Grades'].items():
    # starts a loop that iterates over each subject and its corresponding grade
    # in the grades dictionary of the report card.
        print(f"{subject}: {grade}")
    print(f"Average Grade: {report_card['Average']}")
    print() #adds a newline for formatting.
```

Assignment 9:

Problem Statement:

You are given a list of employees, where each employee is represented as
a dictionary with 'name', 'salary', and 'department' keys.
Write a program to perform the following tasks:

Calculate the total salary for each department.
Find the department with the highest total salary.
Identify the highest paid employee in each department.

#Sample Input:

List of employees
employees = [
{'name': 'John', 'salary': 50000, 'department': 'Sales'},
{'name': 'Jane', 'salary': 60000, 'department': 'Sales'},
{'name': 'Tom', 'salary': 55000, 'department': 'Marketing'},
{'name': 'Emily', 'salary': 70000, 'department': 'Marketing'},
{'name': 'Sam', 'salary': 65000, 'department': 'HR'},
{'name': 'Alex', 'salary': 75000, 'department': 'HR'},
{'name': 'Sarah', 'salary': 60000, 'department': 'IT'},
{'name': 'Michael', 'salary': 80000, 'department': 'IT'},
{'name': 'Jessica', 'salary': 70000, 'department': 'Sales'}
]

#Sample Output:

Total salary for each department:

Sales: 180000

Marketing: 125000

HR: 140000

IT: 140000

Department with the highest total salary: Sales

Highest paid employee in each department:

Sales: Jane (Salary: 70000)

Marketing: Emily (Salary: 70000)

HR: Alex (Salary: 75000)

IT: Michael (Salary: 80000)

List of employees

```
employees = [
    {'name': 'John', 'salary': 50000, 'department': 'Sales'},
    {'name': 'Jane', 'salary': 60000, 'department': 'Sales'},
    {'name': 'Tom', 'salary': 55000, 'department': 'Marketing'},
    {'name': 'Emily', 'salary': 70000, 'department': 'Marketing'},
    {'name': 'Sam', 'salary': 65000, 'department': 'HR'},
    {'name': 'Alex', 'salary': 75000, 'department': 'HR'},
```

```
    {'name': 'Sarah', 'salary': 60000, 'department': 'IT'},
    {'name': 'Michael', 'salary': 80000, 'department': 'IT'},
    {'name': 'Jessica', 'salary': 70000, 'department': 'Sales'}
]

# Task 1: Calculate the total salary for each department
department_salaries = {}

for employee in employees:
    department = employee['department']
    salary = employee['salary']
    department_salaries[department] = department_salaries.get(department, 0)
+ salary

# Task 2: Find the department with the highest total salary
# highest_salary_department = max(department_salaries,
key=department_salaries.get)

highest_salary_department = None
highest_salary = -1

for department, total_salary in department_salaries.items():
    if total_salary > highest_salary:
        highest_salary = total_salary
        highest_salary_department = department

# Printing the department with the highest total salary
```

```
# print(f"\nDepartment with the highest total salary:
{highest_salary_department}")


# Task 3: Identify the highest paid employee in each department

highest_paid_employees = {}


for employee in employees:

    department = employee['department']

    salary = employee['salary']

    if department not in highest_paid_employees or salary >
highest_paid_employees[department]['salary']:

        highest_paid_employees[department] = {'name': employee['name'],
'salary': salary}


# Printing results

print("Total salary for each department:")

for department, total_salary in department_salaries.items():

    print(f"{department}: {total_salary}")


print(f"\nDepartment with the highest total salary:
{highest_salary_department}")


print("\nHighest paid employee in each department:")

for department, employee_info in highest_paid_employees.items():

    print(f"{department}: {employee_info['name']} (Salary:
{employee_info['salary']})")
```

Assignment 10:

Problem Statement:

You are given a list of patients, where each patient is represented as a dictionary

with 'name', 'amount_to_pay', and 'disease' keys.

Write a program to perform the following tasks:

Calculate the total amount to be paid by all patients.

Find the patient with the highest amount to be paid.

Identify the patients suffering from a specific disease.

#Sample Input -

List of patients

patients = [

{'name': 'John', 'amount_to_pay': 500, 'disease': 'Flu'},

{'name': 'Jane', 'amount_to_pay': 1000, 'disease': 'Fever'},

{'name': 'Tom', 'amount_to_pay': 800, 'disease': 'Flu'},

{'name': 'Emily', 'amount_to_pay': 1200, 'disease': 'Covid'},

{'name': 'Sam', 'amount_to_pay': 1500, 'disease': 'Covid'},

{'name': 'Alex', 'amount_to_pay': 700, 'disease': 'Fever'},

{'name': 'Sarah', 'amount_to_pay': 900, 'disease': 'Flu'},

{'name': 'Michael', 'amount_to_pay': 1100, 'disease': 'Covid'},

{'name': 'Jessica', 'amount_to_pay': 600, 'disease': 'Fever'}

]

#Sample Output -

Total amount to be paid by all patients: 8300

Patient with the highest amount to be paid: Sam (Amount: 1500)

Patients suffering from Covid:

Name: Emily, Amount to be paid: 1200

Name: Sam, Amount to be paid: 1500

Name: Michael, Amount to be paid: 1100

```
# List of patients
patients = [
    {'name': 'John', 'amount_to_pay': 500, 'disease': 'Flu'},
    {'name': 'Jane', 'amount_to_pay': 1000, 'disease': 'Fever'},
    {'name': 'Tom', 'amount_to_pay': 800, 'disease': 'Flu'},
    {'name': 'Emily', 'amount_to_pay': 1200, 'disease': 'Covid'},
    {'name': 'Sam', 'amount_to_pay': 1500, 'disease': 'Covid'},
    {'name': 'Alex', 'amount_to_pay': 700, 'disease': 'Fever'},
    {'name': 'Sarah', 'amount_to_pay': 900, 'disease': 'Flu'},
    {'name': 'Michael', 'amount_to_pay': 1100, 'disease': 'Covid'},
    {'name': 'Jessica', 'amount_to_pay': 600, 'disease': 'Fever'}
]

# Task 1: Calculate the total amount to be paid by all patients
total_amount_to_pay = 0
```

```
for patient in patients:
    total_amount_to_pay += patient['amount_to_pay']

# Task 2: Find the patient with the highest amount to be paid
highest_amount_patient = None
highest_amount = -1

for patient in patients:
    if patient['amount_to_pay'] > highest_amount:
        highest_amount = patient['amount_to_pay']
        highest_amount_patient = patient

# Task 3: Identify the patients suffering from a specific disease
specific_disease = 'Covid'
patients_with_specific_disease = []

for patient in patients:
    if patient['disease'] == specific_disease:
        patients_with_specific_disease.append(patient)

# Printing results
print("Total amount to be paid by all patients:", total_amount_to_pay)

print(f"\nPatient with the highest amount to be paid:
{highest_amount_patient['name']} "
      f"(Amount: {highest_amount_patient['amount_to_pay']})")
```

```
print(f"\nPatients suffering from {specific_disease}:")

for patient in patients_with_specific_disease:

    print(f"Name: {patient['name']}, Amount to be paid:
{patient['amount_to_pay']}")
```