Inverting keys and values in a dictionary

In [ ]:
```
def dic_k(dici):
  return {v : k for k, v in dici.items()}

sample_dict = {'a': 3, 'b': 4, 'c': 10, 'd': 5, 'e': 12}
print(dic_k(sample_dict))
```

```
{3: 'a', 4: 'b', 10: 'c', 5: 'd', 12: 'e'}
```

Question 1

In [ ]:
```
# Write a Python program that takes a string from the user, removes all vowels
# remaining letters, and sorts them in alphabetical order. The program should
# a list.

def remove_vowels_and_sort(string):
    vowels = "aeiouAEIOU"
    result = [char for char in string if char not in vowels]
    result.sort()
    return result

user_input = input("Enter a string: ")
sorted_letters = remove_vowels_and_sort(user_input)
print(sorted_letters)
```

```
Enter a string: Vinayak
['V', 'k', 'n', 'y']
```

Question 2

In [ ]:
```
# Write a Python program that takes a list of words from the user, identifies
# are palindromes (words that read the same backward as forward), and then sor
# reverse alphabetical order. The program should finally print the updated lis

def remove_palindromes_and_sort(words):
    non_palindromes = [word for word in words if word != word[::-1]]
    # non_palindromes.sort(reverse=True)
    return tuple(non_palindromes)

words = ['level', 'world', 'civic', 'python', 'madam', 'programming']
updated_list = remove_palindromes_and_sort(words)
print(updated_list)
```

```
('world', 'python', 'programming')
```

Question 3

In [ ]:
```
# Write a Python program that takes two strings as input from the user. The pr
# following operations:
# - Concatenate the two strings.
# - Remove all digits from the concatenated string.
# - Convert the string to uppercase.
# - Print the final result.

def process_strings(str1, str2):
    concatenated = str1 + str2
    no_digits = ''.join([char for char in concatenated if char.isalpha()])
    upper_case = no_digits.upper()
    return upper_case

str1 = input("Enter the first string: ")
str2 = input("Enter the second string: ")
result = process_strings(str1, str2)
print(result)
```

```
Enter the first string: vinay
Enter the second string: 2ak
VINAYAK
```

Question 4

In [3]:

```
# Write a Python program that takes a sentence from the user and prints the se
# example, "Hello World" should be printed as "World Hello".

def reverse_sentence(sentence):
    words = sentence.split()
    reversed_sentence = ' '.join(word for word in words[::-1])
    return reversed_sentence

sentence = input("Enter a sentence: ")
print(reverse_sentence(sentence))
```

```
Enter a sentence: vinayak my name is
isnamemyvinayak
```

Question 5

In [ ]:

```
# Write a Python program that takes a dictionary where the keys are strings an
# Your program should return a new dictionary containing only the key-value pa
# even number.

def filter_even_values(dictionary):
    return {k: v for k, v in dictionary.items() if v % 2 == 0}

sample_dict = {'a': 3, 'b': 4, 'c': 10, 'd': 5, 'e': 12}
filtered_dict = filter_even_values(sample_dict)
print(filtered_dict)
```

```
{'b': 4, 'c': 10, 'e': 12}
```

Question 6

In [ ]:

```
# Write a Python program to count the number of strings where the string lengt
# and last character are the same from a given list of strings.

def count_special_strings(strings):
    return sum(1 for s in strings if len(s) >= 2 and s[0] == s[-1])

sample_list = ['abc', 'xyz', 'aba', '1221']
result = count_special_strings(sample_list)
print(result)
```

```
2
```

Question 7

In [ ]:
```
# Write a Python program that filters a list of integers and counts how many n
# and even.

def count_greater_than_10_and_even(numbers):
    return sum(1 for num in numbers if num > 10 and num % 2 == 0)

sample_list = [12, 7, 15, 22, 10, 6]
result = count_greater_than_10_and_even(sample_list)
print(result)
```

```
2
```

Question 8

In [4]:
```
# Write a program to prepare a grocery bill. For that, enter the name of items
# it is purchased, and its price per unit. Then display the bill in the follow

def print_grocery_bill(dicti):
    print("********************BILL********************")
    print("Item Name\tItem Quantity\tItem Price")
    print("********************************************")
    total_amount = 0
    for item, details in dicti.items():
        print(f"{item}\t\t{details['quantity']}\t\t{details['price']}")
        total_amount += details['quantity'] * details['price']
    print("********************************************")
    print(f"Total Amount to be paid: {total_amount}")
    print("********************************************")

items = {
    'Apple': {'quantity': 2, 'price': 150},
    'Banana': {'quantity': 1, 'price': 40}
}
print_grocery_bill(items)
```

```
********************BILL********************
Item Name        Item Quantity        Item Price
********************************************
Apple                   2                 150
Banana                   1                 40
********************************************
Total Amount to be paid: 340
********************************************
```

Question 9

```
In [ ]: # Write a Python program that takes a list of integers from the user, removes
        # then sorts the list in ascending order. The program should finally convert t

        def remove_duplicates_and_sort(numbers):
            unique_numbers = list(set(numbers))
            unique_numbers.sort()
            return tuple(unique_numbers)

        numbers = [int(x) for x in input("Enter numbers separated by space: ").split()
        result = remove_duplicates_and_sort(numbers)
        print(result)
```

```
Enter numbers separated by space: 1 23 4 5 66 79 69 420 66 23
(1, 4, 5, 23, 66, 69, 79, 420)
```

Question 10

```
In [ ]: # Question 10:
        # Write a Python program to manage student grades. The program should:
        # - Prompt the user to enter student names (as strings) and their correspondin
        # - Store the student names as keys and grades as values in a dictionary.
        # - Calculate and print the average grade of the class.
        # - Identify and print the name of the student with the highest grade.
        # - Convert all student names to uppercase before storing them in the dictiona

        def manage_student_grades():
            students = {}
            for _ in range(5):
                name = input("Enter student name: ").upper()
                grade = int(input(f"Enter grade for {name}: "))
                students[name] = grade

            # Calculate the average grade
            average_grade = sum(students.values()) / len(students)
            print(f"Average grade of the class: {average_grade:.2f}")

            # Identify the student with the highest grade
            highest_grade_student = max(students, key=students.get)
            print(f"Student with the highest grade: {highest_grade_student} ({students

        # Run the program
        manage_student_grades()
```

```
Enter student name: Vinayak
Enter grade for VINAYAK: 69
Enter student name: me
Enter grade for ME: 420
Enter student name: notme
Enter grade for NOTME: 600
Enter student name: notnotme
Enter grade for NOTNOTME: 60
Enter student name: sonotme
Enter grade for SONOTME: 90
Average grade of the class: 247.80
Student with the highest grade: NOTME (600)
```

Question 11

In [ ]:
```python
# Write a program which accepts a sequence of comma-separated 4-digit binary n
# then checks whether they are divisible by 5 or not. The numbers that are div
# a comma-separated sequence.

def check_divisibility_by_5(binary_numbers):
    divisible_by_5 = [num for num in binary_numbers if int(num, 2) % 5 == 0]
    return ', '.join(divisible_by_5)

binary_numbers = input("Enter 4-digit binary numbers separated by commas: ").s
result = check_divisibility_by_5(binary_numbers)
print(result)
```

```
Enter 4-digit binary numbers separated by commas: 1010,1111,1000,0001
1010, 1111
```

Question 12

In [ ]:
```python
# Write a program to calculate the final price of a product after applying a d
# arithmetic and assignment operators, and round the final result to two decim

def calculate_final_price(price, discount, tax):
    price_after_discount = price - (price * discount / 100)
    final_price = price_after_discount + (price_after_discount * tax / 100)
    return round(final_price, 2)

price = float(input("Enter the price of the product: "))
discount = float(input("Enter the discount percentage: "))
tax = float(input("Enter the tax percentage: "))
final_price = calculate_final_price(price, discount, tax)
print(f"Final price: {final_price}")
```

Question 13

In [ ]:
```python
# Write a Python program to find out if the letter 'A' is available or not in

def count_letter_a(string):
    count = string.upper().count('A')
    return count > 0, count
```

```
str1 = 'KJSCE'
is_present, count = count_letter_a(str1)
print(f"Is 'A' present: {is_present}")
print(f"Count of 'A': {count}")
```

Question 14

```
In [ ]: # Create a dictionary of products purchased and their MRPs. Calculate the bill

def calculate_bill(products, quantities):
    total_amount = sum(products[item] * quantities[item] for item in quantitie
    return total_amount

products = {"Apple": 150, "Banana": 40, "Cherry": 200, "Date": 250}
quantities = {"Apple": 2, "Banana": 1}
total_bill = calculate_bill(products, quantities)
print(f"Total bill: {total_bill}")
```

```
Total bill: 340
```

Question 15

```
In [ ]: # Question 15:
# (a) Write a program to input a decimal number and convert it to binary using
# (b) Extend the above program to convert the binary number back to decimal.

def decimal_to_binary(decimal_number):
    binary_number = ""
    while decimal_number > 0:
        remainder = decimal_number % 2
        binary_number = str(remainder) + binary_number
        decimal_number = decimal_number // 2
    return binary_number

def binary_to_decimal(binary_number):
    decimal_number = 0
    binary_number = binary_number[::-1]  # Reverse the binary string
    for i in range(len(binary_number)):
        decimal_number += int(binary_number[i]) * (2 ** i)
    return decimal_number

# Input from the user
decimal_number = int(input("Enter a decimal number: "))
binary_number = decimal_to_binary(decimal_number)
print(f"Binary representation of {decimal_number} is {binary_number}")

# Convert the binary number back to decimal
decimal_number_converted_back = binary_to_decimal(binary_number)
print(f"Decimal representation of {binary_number} is {decimal_number_converted
```

Question 16

In [ ]:
```
# Write a program to calculate the sum of the marks and display average marks
# for the 5 subjects. The maximum marks of each subject is 100. Read the subje
# marks. Follows with computation of sum of marks. Finally display the name of
# subject marks and average marks obtained in the given exam.

def calculate_marks():
    student_name = input("Enter the student's name: ")
    subjects = {}
    total_marks = 0

    for _ in range(5):
        subject = input("Enter the subject name: ")
        marks = float(input(f"Enter the marks for {subject}: "))
        subjects[subject] = marks
        total_marks += marks

    average_marks = total_marks / 5

    print(f"Student Name: {student_name}")
    print("Subject-wise Marks:")
    for subject, marks in subjects.items():
        print(f"{subject}: {marks}")
    print(f"Total Marks: {total_marks}")
    print(f"Average Marks: {average_marks:.2f}")

calculate_marks()
```

Question 17

In [ ]:
```
# Write a program to calculate a student’s result based on two examinations, 1
# conducted. The weightage of activities = 30 percent, sports = 20 percent, ar

def calculate_student_result():
    exam1 = float(input("Enter the marks for the first examination: "))
    exam2 = float(input("Enter the marks for the second examination: "))
    sports = float(input("Enter the marks for the sports event: "))
    activities = [float(input(f"Enter the marks for activity {i+1}: ")) for i

    exam_weightage = 0.5
    sports_weightage = 0.2
    activities_weightage = 0.3

    exam_average = (exam1 + exam2) / 2
    activities_average = sum(activities) / len(activities)

    final_result = (exam_average * exam_weightage) + (sports * sports_weightag

    print(f"Final Result: {final_result:.2f}")

calculate_student_result()
```

```
Enter the marks for the first examination: 20
Enter the marks for the second examination: 20
Enter the marks for the sports event: 30
Enter the marks for activity 1: 3
Enter the marks for activity 2: 4
Enter the marks for activity 3: 5
Final Result: 17.20
```

Question 18

In [ ]:
```
# Write a Python program that takes a list of student names and their correspo
# information in a dictionary, and then prints out the names of students who h
# threshold.

def filter_students_by_grade(threshold):
    students = {}
    for _ in range(5):
        name = input("Enter student name: ")
        grade = int(input("Enter student grade: "))
        students[name] = grade

    filtered_students = {name: grade for name, grade in students.items() if gr

    print("Students with grades above the threshold:")
    for name, grade in filtered_students.items():
        print(f"{name}: {grade}")

threshold = int(input("Enter the grade threshold: "))
filter_students_by_grade(threshold)
```

```
Enter the grade threshold: 45
Enter student name: Vinayak
Enter student grade: 75
Enter student name: me
Enter student grade: 55
Enter student name: notme
Enter student grade: 44
Enter student name: meme
Enter student grade: 45
Enter student name: memememe
Enter student grade: 45
Students with grades above the threshold:
Vinayak: 75
me: 55
```

Question 19

In [ ]:
```
# Write a Python program to perform the following operation on a list of numbe
# - Create a list of 3 odd numbers
# - Create a list of 3 even numbers
# - Combine the two lists
# - Add prime numbers 11, 17, 29 at the beginning of the combined list
# - Report how many elements are present in the list
# - Delete the list

def list_operations():
    odd_numbers = [1, 3, 5]
    even_numbers = [2, 4, 6]
    combined_list = odd_numbers + even_numbers
    prime_numbers = [11, 17, 29]
    combined_list = prime_numbers + combined_list

    print(f"Combined List: {combined_list}")
    print(f"Number of elements in the list: {len(combined_list)}")

    del combined_list
    print("List deleted.")

list_operations()
```

Question 20

```
In [ ]: # Write a Python program that:
# - Gets the current date and time.
# - Formats it into a human-readable string (e.g., "YYYY-MM-DD HH:MM").
# - Calculates the number of days between today and a user-provided date.

from datetime import datetime

def date_operations():
    current_datetime = datetime.now()
    formatted_datetime = current_datetime.strftime("%Y-%m-%d %H:%M")
    print(f"Current Date and Time: {formatted_datetime}")

    user_date = input("Enter a date (YYYY-MM-DD): ")
    user_date = datetime.strptime(user_date, "%Y-%m-%d")
    days_difference = (current_datetime - user_date).days

    print(f"Number of days between today and the user-provided date: {days_dif

date_operations()
```

```
Current Date and Time: 2024-11-26 13:41
Enter a date (YYYY-MM-DD): 2006-09-03
Number of days between today and the user-provided date: 6659
```

Question 22

```
In [ ]: # Write a Python program to find out if vowels in a given string are present c
# each vowel is present.

def count_vowels(string):
    vowels = "aeiouAEIOU"
    vowel_count = {vowel: string.count(vowel) for vowel in vowels if vowel in
    return bool(vowel_count), vowel_count
```

```
str1 = 'ABCDE'
is_present, count = count_vowels(str1)
print(f"Are vowels present: {is_present}")
print(f"Vowel counts: {count}")
```

Question 23

```
In [ ]: # Write a program to display the digit at the one's place of a number.

        def ones_place_digit(number):
            return abs(number) % 10

        number = int(input("Enter a number: "))
        print(f"The digit at the one's place is: {ones_place_digit(number)}")
```

```
Enter a number: 101
The digit at the one's place is: 1
```

Question 24

```
In [ ]: # Write a Python program that takes two tuples test_tuple1 and test_tuple2 and
        # combinations of the two tuples.

        def pair_combinations(tuple1, tuple2):
            return [(a, b) for a in tuple1 for b in tuple2]

        test_tuple1 = (7, 2)
        test_tuple2 = (7, 8)
        result = pair_combinations(test_tuple1, test_tuple2)
        print(result)
```

```
[(7, 7), (7, 8), (2, 7), (2, 8)]
```

Question 25

```
In [ ]: # Write a Python program to reverse the order of the items in the array.

        # def reverse_array(arr):
        #     return arr[::-1]

        # arr = array('i', [1, 3, 5, 3, 7, 1, 9, 3])
        arr = [1, 3, 5, 3, 7, 1, 9, 3]
        reversed_arr = arr[::-1]
        print(reversed_arr)
```

```
[3, 9, 1, 7, 3, 5, 3, 1]
```

Question 26

```
In [ ]: # Write a Python program to count the number of characters in a string.

        def count_characters(string):
            return {char: string.count(char) for char in set(string)}
```

```
sample_string = 'google.com'
result = count_characters(sample_string)
print(result)
```

```
{'m': 1, '.': 1, 'e': 1, 'g': 2, 'o': 3, 'l': 1, 'c': 1}
```

Question 27

```
In [ ]: # Write a Python program to remove characters from a string starting from zerc
        # string.

        def remove_characters(string, n):
            return string[n:]

        sample_string = 'Somaiya University'
        n = 5
        new_string = remove_characters(sample_string, n)
        print(new_string)
```

```
ya University
```

Question 28

```
In [ ]: # Write a program to remove tuples from a list of tuples if the sum of the ele
        # than n.

        def remove_tuples(tuples_list, n):
            return [t for t in tuples_list if sum(t) <= n]

        sample_list = [(1, 2), (3, 4), (5, 6), (7, 8)]
        n = 10
        filtered_list = remove_tuples(sample_list, n)
        print(filtered_list)
```

Question 29

```
In [ ]: # Identify the errors in the following code and provide the corrections:
        # Reading a string from the user
        user_input = input("Enter a string: ")

        # Splitting the string into words
        words = user_input.split()

        # Converting each word to uppercase
        uppercase_words = list(map(str.upper, words))

        # Printing the original and uppercase lists
        print("Original words:", words)
        print("Uppercase words:", uppercase_words)
```

```
Enter a string: Vinayak pai is graet
Original words: ['Vinayak', 'pai', 'is', 'graet']
Uppercase words: ['VINAYAK', 'PAI', 'IS', 'GRAET']
```

Question 30

```
In [ ]: # Write a Python program to count the number of strings in a given list where
        # and the string contains the same character at least twice.

        def count_special_strings(strings):
            return sum(1 for s in strings if len(s) >= 3 and any(s.count(char) > 1 for

        sample_list = ['hello', 'world', 'noon', 'abcd', 'aabbcc']
        result = count_special_strings(sample_list)
        print(result)
```

Question 31

```
In [ ]: # Write a Python program that takes a list of dictionaries, where each diction
        # their name and scores in various subjects. The program should return a new d
        # a student's name, and the value is their average score across all subjects.

        def calculate_average_scores(students):
            averages = {}
            for student in students:
                name = student.pop('name')
                average_score = sum(student.values()) / len(student)
                averages[name] = round(average_score, 2)
            return averages

        students = [
            {'name': 'Arjun', 'math': 85, 'science': 90, 'english': 78},
            {'name': 'Balram', 'math': 92, 'science': 88, 'english': 84},
            {'name': 'Damodar', 'math': 72, 'science': 75, 'english': 80}
        ]
        result = calculate_average_scores(students)

        print(result)
```

```
[{'math': 85, 'science': 90, 'english': 78}, {'math': 92, 'science': 88, 'engli
sh': 84}, {'math': 72, 'science': 75, 'english': 80}]
{'Arjun': 84.33, 'Balram': 88.0, 'Damodar': 75.67}
```

Question 32

```
In [ ]: # Perform slice operations on the given string to get the following output.
        # Input String: "Python is Easy!"
        # Output:
        # - P
        # - on is Easy
        # - !
        # - Python is Easy!Python is Easy!
        # - Python is Easy!Isn't it?

        input_string = "Python is Easy!"

        # Slicing operations
```

```
print(input_string[0])  # P
print(input_string[4:])  # on is Easy
print(input_string[-1])  # !
print(input_string * 1)  # Python is Easy!Python is Easy!
print(input_string + "Isn't it?")  # Python is Easy!Isn't it?
```

```
P
on is Easy!
!

Python is Easy!Isn't it?
```

Question 33

```
In [ ]: # Create an empty dictionary. Then add the {digit: ASCII value} pairs for all
        # i.e., {0: 48}, {1: 49}, etc., to this dictionary and print the dictionary. T
        # use the ord function, i.e., ord('0') = 48.

        ascii_dict = {}
        for digit in range(10):
            ascii_dict[digit] = ord(str(digit))

        # digit_ascii = {v: k for k, v in ascii_dict.items()}

        # print(digit_ascii)

        print(ascii_dict)
```

```
{48: 0, 49: 1, 50: 2, 51: 3, 52: 4, 53: 5, 54: 6, 55: 7, 56: 8, 57: 9}
{0: 48, 1: 49, 2: 50, 3: 51, 4: 52, 5: 53, 6: 54, 7: 55, 8: 56, 9: 57}
```