Python Programming Practical DCA-3132

1. **Write a Program to find the largest number among three numbers . Take input/output as specified – Print the expected output using the expected logic/ algorithm / data – code is structured correctly and according to the problem statement .**

Solution 1 :-

```
# Input three numbers from the user
num1 = float(input("Enter the first number: "))
num2 = float(input("Enter the second number: "))
num3 = float(input("Enter the third number: "))

# Compare the numbers to find the largest
if num1 >= num2 and num1 >= num3:
   largest = num1
elif num2 >= num1 and num2 >= num3:
   largest = num2
else:
   largest = num3

# Print the largest number
print("The largest number is: {}".format( largest) )
```

Program Output

```
Enter the first number: 1
Enter the second number: 88
Enter the third number: 74
The largest number is: 88
```

2. Write a Program to print all prime number of particular interval . (like prime numbers between 2 and 20) Take input/output as specified – Print the expected output using the expected logic/ algorithm / data – code is structured correctly and according to the problem statement .

Solution 2 :-

```
# Function to check if a number is prime
def is_prime(num):
    if num <= 1:
        return False
    if num <= 3:
        return True
    if num % 2 == 0 or num % 3 == 0:
        return False
    i = 5
    while i * i <= num:
        if num % i == 0 or num % (i + 2) == 0:
            return False
        i += 6
    return True

# Input the interval from the user
start = int(input("Enter the starting number of the interval: "))
end = int(input("Enter the ending number of the interval: "))

# Validate input
if start >= end or start < 2:
    print("Invalid input. Please enter a valid interval.")
else:
    # Print prime numbers within the interval
    print("Prime numbers between {} and {} are:".format(start, end))
    for num in range(start, end + 1):
        if is_prime(num):
            print(num, end=" ")
```

Program Output

```
Enter the starting number of the interval: 2
Enter the ending number of the interval: 20
Prime numbers between 2 and 20 are: 2  3  5  7  11  13  17  19
```

3. Write a Program to print alist having duplicates from a list of integers [1,2,3,4,5,1,1,2,5,6,7,8,9,]. Take input/output as specified – Print the expected output using the expected logic/ algorithm / data – code is structured correctly and according to the problem statement .

Solution 3 :-

```
# Input list of integers
input_list = [1, 2, 3, 4, 5, 1, 1, 2, 5, 6, 7, 8, 9]

# Initialize an empty list to store duplicate integers
duplicate_list = []

# Create a dictionary to count the occurrences of each integer
count_dict = {}

# Iterate through the input list
for num in input_list:
    # If the number is already in the dictionary, increment its count
    if num in count_dict:
        count_dict[num] += 1
    else:
        count_dict[num] = 1

# Iterate through the dictionary and add numbers with counts > 1 to the duplicate list
for num, count in count_dict.items():
    if count > 1:
        duplicate_list.append(num)

# Print the list of duplicate integers
print("List of duplicate integers:", duplicate_list)
```

Program Output

List of duplicate integers: [1, 2, 5]

4. **Write a Program to Display Fibonacci Sequence Using Recursion . Take input/output as specified – Print the expected output using the expected logic/ algorithm / data – code is structured correctly and according to the problem statement .**

Solution 3 :-

```
# Function to generate Fibonacci sequence using recursion
def fibonacci(n):
    if n <= 0:
        return []
    elif n == 1:
        return [0]
    elif n == 2:
        return [0, 1]
    else:
        fib_seq = fibonacci(n - 1)
        fib_seq.append(fib_seq[-1] + fib_seq[-2])
        return fib_seq

# Input the number of terms in the sequence from the user
n = int(input("Enter the number of terms in the Fibonacci sequence: "))

# Validate input
if n <= 0:
    print("Please enter a positive integer.")
else:
    # Generate and print the Fibonacci sequence
    fib_sequence = fibonacci(n)
    print("Fibonacci Sequence (first {} terms):".format(n))
    for term in fib_sequence:
        print(term, end=" ")
```

Program Output

```
Enter the number of terms in the Fibonacci sequence: 10
Fibonacci Sequence (first 10 terms): 0  1  1  2  3  5  8  13  21  34
```

5. Write a Program to generate dictionary of frequency of alphabets of given string : “Online_Manipal” - Take input/output as specified – Print the expected output using the expected logic/ algorithm / data – code is structured correctly and according to the problem statement .

Solution 3 :-

```
# Input the string
input_string = "Online_Manipal"
# Initialize an empty dictionary to store the frequency of alphabets
alphabet_frequency = {}

# Convert the string to lowercase (optional, to treat uppercase and lowercase letters as the same)
input_string = input_string.lower()

# Iterate through the characters in the string
for char in input_string:
    # Check if the character is an alphabet (a-z)
    if char.isalpha():
        # If the alphabet is already in the dictionary, increment its count
        if char in alphabet_frequency:
            alphabet_frequency[char] += 1
        # If the alphabet is not in the dictionary, add it with a count of 1
        else:
            alphabet_frequency[char] = 1

# Print the dictionary of alphabet frequencies
print("Dictionary of Alphabet Frequencies:")
for alphabet, frequency in alphabet_frequency.items():
    print(f"{alphabet}: {frequency}")
```

Program Output

```
Dictionary of Alphabet Frequencies:
O: 1
n: 3
l: 2
i: 2
e: 1
m: 1
a: 2
p: 1
```