Q-1→ Write a program to determine the type of input using match statement.

```
def determine_type(input):
    match input:
        case int():
            return "This is an integer."
        case float():
            return "This is a float."
        case str():
            return "This is a string."
        case bool():
            return "This is a boolean."
        case list():
            return "This is a list."
        case tuple():
            return "This is a tuple."
        case dict():
            return "This is a dictionary."
        case set():
            return "This is a set."
        case _:
            return "Unknown type."

print(determine_type(123))
print(determine_type(123.45))
print(determine_type("Hello, World!"))
print(determine_type(False))
print(determine_type([1, 2, 3]))
print(determine_type((1, 2, 3)))
print(determine_type({"key": "value"}))
print(determine_type({1, 2, 3}))
print(determine_type(object()))
```

Q-2→ Find the factorial of a number using function.

```
def factorial(n):
    result = 1
    for i in range(1, n + 1):
        result *= i
    return result

num = int(input("Enter number: "))
print("Factorial of",num,"is: ",factorial(num))
```

Q-3→ Write a program to print even numbers using continue statement.

```
n= int(input("Enter number: "))
l=list()
for num in range(1,n+1):
    if num % 2 == 0:
        print(f"Even number: {num}")
        continue
    else:
        print()
```

Q-4→Write a program to demonstrate list, set, tuple and dictionary.

```
n=int(input('''Enter
            1 for list
            2 for set
            3 for tuple
            4 for dictionary'''))
if n==1:
    print("List=")
    print([1,2,3,4,5,6])
elif n==2:
    print('Set')
    print({1,2,3,4,5,6})
elif n==3:
    print("tuple")
    t=(1,2,3,4,5)
    print(t)
elif n==4:
    print("Dictionary")
    print({1:2,3:4,5:6})
else:
    print()
```

Q-5→ Write a program to count number of vowels in a string.

```
def count_vowels(string):
    vowels = "aeiouAEIOU"
    return sum(1 for char in string if char in vowels)

# Test cases
string = input("Enter your name")
print(f"Number of vowels in '{string}': {count_vowels(string)}")
```

Q-6→ Write a program to find maximum and minimum of n numbers.

```
def find_max_min():
    n = int(input("Enter the number of elements: "))
    numbers = []
    print("Enter the elements:")
    for i in range(n):
        number = float(input())
        numbers.append(number)
    min_value = min(numbers)
    max_value = max(numbers)
    print(f"Minimum value: {min_value}")
    print(f"Maximum value: {max_value}")

find_max_min()
```

Q-7→ Write a program to check number is prime or not.

```
def is_prime(n):
    if n < 2:
        return False
    return not any(n % i == 0 for i in range(2, int(n**0.5) + 1))

num = int(input("Enter number"))
if is_prime(num):
    print(f"{num} is a prime number.")
else:
    print(f"{num} is not a prime number.")
```

Q-8→ Write a program to display the first occurrence of number divisible by k in the list.

```
def first_occurrence_divisible_by_k(numbers, k):
    return next((i for i, number in enumerate(numbers) if number % k == 0), -1)

numbers = [1, 3, 6, 8, 12, 15, 20]
k = 5
index = first_occurrence_divisible_by_k(numbers, k)
if index != -1:
    print(f"The first occurrence of a number divisible by {k} in the list is at index {index}.")
else:
    print(f"No number in the list is divisible by {k}.")
```

Q-9→ write a program to count occurrence of each element in the list.

```python
def count_occurrences(numbers):
    count = {}
    for number in numbers:
        if number not in count:
            count[number] = 1
        else:
            count[number] += 1
    return count

numbers = [1, 3, 6, 8, 1, 3, 3, 6, 8, 8, 10]
occurrences = count_occurrences(numbers)
print(f"Occurrences of each element in the list: {occurrences}")
```

Q-10→ Check a string is palindrome or not.

```python
def is_palindrome(s):
    s = s.lower()
    s = ''.join(c for c in s if c.isalnum())
    return s == s[::-1]

string1 = "racecar"

if is_palindrome(string1):
    print(f"{string1} is a palindrome.")
else:
    print(f"{string1} is not a palindrome.")
```

Q-11→ Generate Fibonacci sequence up to n terms using while loop.

```python
def fibonacci(n):
    if n <= 0:
        return 0
    elif n == 1:
        return 1
    else:
        a, b = 0, 1
        for _ in range(n - 1):
            a, b = b, a + b
        return a

# Test cases
n = 10
print(f"The {n}th Fibonacci number is: {fibonacci(n)}")
```

Q-12→ Write a program to find all prime numbers from 1 to n numbers.

```python
def is_prime(n):
    if n < 2:
        return False
    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:
            return False
    return True

def find_primes(n):
    primes = []
    for num in range(1, n + 1):
        if is_prime(num):
            primes.append(num)
    return primes

# Test cases
n = 50
primes = find_primes(n)
print(f"Prime numbers from 1 to {n}: {primes}")
```

Q-13→ Write a program to print all the names from list whose length greater than 6.

```python
names = ["Alice", "Bob", "Charlie", "David", "Eve", "Frankyy", "George", "Hannah", "Ivan", "Judy"]
long_names = [name for name in names if len(name) > 6]
print(f"Names with length greater than 6: {long_names}")
```

Q-14→ Find the sum of digit of a number

```python
def sum_of_digits(n):
    if n < 0:
        n = n
    return sum(map(int, str(n)))

# Test cases
n = 12345
print(f"Sum of digits of {n}: {sum_of_digits(n)}")
```

Q-15→ Write a program to check a number is binary

```python
def is_binary(n):
    if n == 0 or n == 1:
        return True
    while n > 0:
        if n % 2 not in (0, 1):
            return False
        n //= 2
    return True

# Test cases
n = 1011
print(f"Is {n} binary? {is_binary(n)}")
n = 10
print(f"Is {n} binary? {is_binary(n)}")
```

Q-16→ Write a program to remove vowels from string

```python
def remove_vowels(s):
    vowels = "aeiouAEIOU"
    return "".join(c for c in s if c not in vowels)

# Test cases
s = "Hello, World!"
print(f"Original string: {s}")
print(f"String without vowels: {remove_vowels(s)}")
```

Q-17→ Write a program to display nth Fibonacci number

```python
def fibonacci(n):
    if n <= 0:
        return 0
    elif n == 1:
        return 1
    else:
        a, b = 0, 1
        for _ in range(n - 1):
            a, b = b, a + b
        return a

# Test cases
n = 10
print(f"The {n}th Fibonacci number is: {fibonacci(n)}")
```

Q-18→ Check a number is Armstrong

```python
def is_armstrong(n):
    n_str = str(n)
    n_len = len(n_str)
    n_sum = sum(int(digit) ** n_len for digit in n_str)
    return n_sum == n

# Test cases
n = 153
print(f"Is {n} an Armstrong number? {is_armstrong(n)}")
n = 154
print(f"Is {n} an Armstrong number? {is_armstrong(n)}")
```

Q-19→ Write a program to print ASCII value of all the characters of string with character

```python
s = "python"
print("Character\tASCII Value")
print("-----\t" + "---")
for c in s:
    print(f"{c}\t{ord(c)}")
```

Q-20→ Write a program to check whether a list is monotonic or not

```python
def is_monotonic(lst):
    if len(lst) <= 1:
        return True
    if lst[0] < lst[-1]:
        return all(x <= y for x, y in zip(lst, lst[1:]))
    elif lst[0] > lst[-1]:
        return all(x >= y for x, y in zip(lst, lst[1:]))
    else:
        return all(x == y for x, y in zip(lst, lst[1:]))

# Test cases
lst1 = [1, 2, 3, 4, 5]
print(f"Is {lst1} monotonic? {is_monotonic(lst1)}")
lst2 = [1, 2, 3, 2, 1]
print(f"Is {lst2} monotonic? {is_monotonic(lst2)}")
lst3 = [1, 2, 2, 1]
print(f"Is {lst3} monotonic? {is_monotonic(lst3)}")
lst4 = [1, 2, 2, 2, 1]
print(f"Is {lst4} monotonic? {is_monotonic(lst4)}")
```

Q-21→ Write a program to check a particular element is present in the array or not

```python
def is_element_present(arr, target):
    return target in arr

# Test cases
arr = [1, 2, 3, 4, 5]
target = 3
print(f"Is {target} present in {arr}? {is_element_present(arr, target)}")
target = 6
print(f"Is {target} present in {arr}? {is_element_present(arr, target)}")
```

Q-22→ Write a program to find nth largest and nth smallest element from list

```python
def nth_largest(lst, n):
    return sorted(lst, reverse=True)[n - 1]

def nth_smallest(lst, n):
    return sorted(lst)[n - 1]

# Test cases
lst = [1, 2, 3, 4, 5]
n = 3
print(f"The {n}th largest element in {lst} is {nth_largest(lst, n)}")
n = 2
print(f"The {n}th smallest element in {lst} is {nth_smallest(lst, n)}")
```

Q-23→ Write a program to remove kth character from string

```python
def remove_kth_char(s, k):
    if k <= 0 or k > len(s):
        return s
    else:
        return s[:k - 1] + s[k:]

# Test cases
s = "python"
k = 3
print(f"The string {s} without the {k}th character is {remove_kth_char(s, k)}")
k = 0
print(f"The string {s} without the {k}th character is {remove_kth_char(s, k)}")
k = 6
print(f"The string {s} without the {k}th character is {remove_kth_char(s, k)}")
```

Q-24→ Check a particular substring present in the string

```python
def is_substring_present(s, sub):
    return sub in s

# Test cases
s = "python"
sub = "py"
print(f"Is {sub} present in {s}? {is_substring_present(s, sub)}")
sub = "tho"
print(f"Is {sub} present in {s}? {is_substring_present(s, sub)}")
sub = "xyz"
print(f"Is {sub} present in {s}? {is_substring_present(s, sub)}")
```

Q-25→ Write a program to print all the characters whose length is even

```python
def even_length_chars(s):
    return list(filter(lambda c: len(c) % 2 == 0, s))

# Test case
s = "python"
print(f"Even length characters in {s} are {even_length_chars(s)}")

s = "hello world"
print(f"Even length characters in {s} are {even_length_chars(s)}")
```

Q-26→ Remove duplicate elements from list

```python
def remove_duplicates(lst):
    return list(dict.fromkeys(lst))

# Test case
lst = [1, 2, 2, 3, 4, 4, 5, 6, 6, 7, 8, 8, 9]
print(f"List with duplicates removed: {remove_duplicates(lst)}")
```

Q-27→ Find words whose length is greater than j and less than k

```python
def find_words(s, j, k):
    words = s.split()
    return [word for word in words if len(word) > j and len(word) < k]

# Test case
s = "This is a test string with some words in it"
j, k = 3, 5
print(f"Words with length greater than {j} and less than {k} are {find_words(s, j, k)}")
```

Q-28→ Check a string is binary or not

```
def is_binary(s):
    return not any(c not in '01' for c in s)

# Test cases
s1 = '011010101'
s2 = '1010101a'
print(f"Is {s1} binary? {is_binary(s1)}")
print(f"Is {s2} binary? {is_binary(s2)}")
```

Q-29→ Find uncommon words from two string

```
def uncommon_words(s1, s2):
    words1 = s1.split()
    words2 = s2.split()
    common = set(words1) & set(words2)
    return list(words1 - common) + list(words2 - common)

# Test case
s1 = 'This is a test string with some words in it'
s2 = 'This is a test string with some different words in it'
print(f"Uncommon words are {uncommon_words(s1, s2)}")
```

Q-30→Check a mail is valid or not

```
def is_valid_email(email):
    return bool(re.match(pattern, email))

# Test cases
email1 = 'example@example.com'
email2 = 'example@example'
email3 = 'example.example@example.com'
email4 = 'example@example.c'
email5 = 'example@.com'
print(f"Is {email1} valid? {is_valid_email(email1)}")
print(f"Is {email2} valid? {is_valid_email(email2)}")
print(f"Is {email3} valid? {is_valid_email(email3)}")
print(f"Is {email4} valid? {is_valid_email(email4)}")
print(f"Is {email5} valid? {is_valid_email(email5)}")
```

Q-31→ Check a mobile number is valid or not

```python
import re

def is_valid_mobile(mobile):
    pattern = r'^(\+\d{1,3}\s?)?((\(\d{3}\)|\d{3})[\s.-]?\d{3}[\s.-]?\d{3,4}$'
    return bool(re.fullmatch(pattern, mobile))

# Test cases
mobile1 = '+1 (123) 456-7890'
mobile2 = '123 456 7890'
mobile3 = '123-456-7890'
mobile4 = '123.456.7890'
mobile5 = '123 456 789'
mobile6 = '1234567890'
mobile7 = '+1 123 456 7890'
print(f"Is {mobile1} valid? {is_valid_mobile(mobile1)}")
print(f"Is {mobile2} valid? {is_valid_mobile(mobile2)}")
print(f"Is {mobile3} valid? {is_valid_mobile(mobile3)}")
print(f"Is {mobile4} valid? {is_valid_mobile(mobile4)}")
print(f"Is {mobile5} valid? {is_valid_mobile(mobile5)}")
print(f"Is {mobile6} valid? {is_valid_mobile(mobile6)}")
print(f"Is {mobile7} valid? {is_valid_mobile(mobile7)}")
```

Q-32→ Display only digits from tuple

```python
def display_digits(tup):
    return tuple(c for c in tup if c.isdigit())

# Test cases
tup1 = (1, 2, 3, 4, 5)
tup2 = (1.0, 2.0, 3, 4, 5)
tup3 = ('1', '2', '3', '4', '5')
tup4 = (1, 2, 3, 'a', 4, 5)
tup5 = (1, 2, 3, '123', 4, 5)
print(f"Digits in {tup1} are {display_digits(tup1)}")
print(f"Digits in {tup2} are {display_digits(tup2)}")
print(f"Digits in {tup3} are {display_digits(tup3)}")
print(f"Digits in {tup4} are {display_digits(tup4)}")
print(f"Digits in {tup5} are {display_digits(tup5)}")
```