Name : K.ASHOK KUMAR REDDY
ID : COMETFWC016
Batch : 2
Date : 6/6/2025

#Exercise 1.Calculate the multiplication and sum of two numbers

```
def calculate_sum_and_multiply(num1, num2):
    sum_result = num1 + num2
    product_result = num1 * num2
    return sum_result, product_result

number1 = float(input("Enter the first number: "))
number2 = float(input("Enter the second number: "))

# Calculate sum and multiplication
sum_val, product_val = calculate_sum_and_multiply(number1, number2)

# Display the results
print(f"Sum of {number1} and {number2} is: {sum_val}")
print(f"Multiplication of {number1} and {number2} is: {product_val}")
```

#Exercise 2.writing python program to Print the sum of a current number and a previous number

```
numbers = input("Enter numbers separated by spaces: ").split()
numbers = [int(num) for num in numbers]

previous = 0  # Start with previous number as 0

for current in numbers:
    total = current + previous  # Sum current and previous numbers
    print(f"Sum of current number {current} and previous number {previous} is: {total}")
    previous = current  # Update previous number to current number for next loop
```

#Exercise 3.printing characters present at even index number using python program

```
# Get input string from user
text = input("Enter a string: ")

# Loop through the string, stepping by 2 to get even indices
even_index_chars = ""
for i in range(0, len(text), 2):
    even_index_chars += text[i]

print("Characters at even indices:", even_index_chars)
```

#Exercise 4.Remove first n characters from astring using python program

```
# Get input string from the user
text = input("Enter a string: ")

# Get number of characters to remove
n = int(input("Enter number of characters to remove from the start: "))

# Remove first n characters using slicing
result = text[n:]

print("String after removing first", n, "characters:", result)
```

#Exercise 5.check if the First and Last Numbers of a list are the same using python program

```
# Get a list of numbers from user input
numbers = input("Enter numbers separated by spaces: ").split()
numbers = [int(num) for num in numbers]

# Check if the list is empty first
if len(numbers) == 0:
    print("The list is empty.")
else:
    # Compare first and last elements
    if numbers[0] == numbers[-1]:
        print("Yes, the first and last numbers are the same.")
    else:
        print("No, the first and last numbers are different.")
```

#Exercise 6.DIsplay numbers divisible by 5 using python program

```
# Get list of numbers from user
numbers = input("Enter numbers separated by spaces: ").split()
numbers = [int(num) for num in numbers]

print("Numbers divisible by 5:")

for num in numbers:
    if num % 5 == 0:
        print(num)
```

#Exercise 7.FInd the number of occurences of a substring in a string using python program

```
# Get input string and substring from user
main_string = input("Enter the main string: ")
substring = input("Enter the substring to find: ")

# Count occurrences of substring in main string
count = main_string.count(substring)

print(f"The substring '{substring}' occurs {count} times in the given string.")
```

#1.Write a program to print " hello world " on the screen

```
print("Hello World");
```

#2. Write a program to add two numbers and display the sum.

```
a=int(input("enter the first num:"));
b=int(input("enter the second num:"));
sum=a+b
print("the sum of two numbers is:",sum);
```

3. Write a program to find the largest of three numbers.

```
num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))
num3 = float(input("Enter third number: "))

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

# Output the result
print("The largest number is:", largest)
```

#4. Write a program to check whether a given number is even or odd.

```
a=int(input("enter the number"));
if(a%2==0):
        print("the number is even");
elif(a=0):
        print("zero is neither even nor odd");
```

```
else:
        print("the number is odd");
```

#5. Write a program to find the sum of all the numbers in a list.

```
list=[10,20,30,40,50];
total=sum(list)
print("the su of numbers in the list:",total);
```

#6. Write a program to print the Fibonacci series up to a given number.

```
limit=int(input("enter the upper limit of the fabinocci series:"));
a,b=0,1
print("the fabinocci series upto",limit,":");
while(a<=limit):
    print(a,end=" ");
    a, b=b, a+b
```

#7. Write a program to convert Celsius to Fahrenheit.

```
c=int(input("enter the temp in celsius:"));
f=((9/5)*c)+32
print("the temp in faurenheit is :",f);
```

#8. Write a program to find the factorial of a number.

```
num = int(input("Enter a number: "))

# Check if the number is negative
if num < 0:
    print("Factorial does not exist for negative numbers.")
elif num == 0:
    print("The factorial of 0 is 1.")
else:
    factorial = 1
    for i in range(1, num + 1):
        factorial *= i
    print("The factorial of", num, "is", factorial)
```

#9. Write a program to check whether a given number is prime or not.

```
# Program to check if a number is prime

num = int(input("Enter a number: "))

if num <= 1:
   print(num, "is not a prime number.")
else:
   for i in range(2, num):
      if num % i == 0:
         print(num, "is not a prime number.")
         break
   else:
      print(num, "is a prime number.")
```

#10. Write a program to find the GCD of two numbers.

```
def find_gcd(a, b):
   while b:
      a, b = b, a % b
   return a

# Input from user
num1 = int(input("Enter the first number: "))
num2 = int(input("Enter the second number: "))

gcd = find_gcd(num1, num2)
print(f"The GCD of {num1} and {num2} is {gcd}.")
```

#11. Write a program to reverse a string.

```
string=input("enter the given string:");
reversed_string=string[::-1]
print("the reversed string is:",reversed_string);
```

#12. Write a program to find the sum of the digits of a given number.

```
def sum_of_digits(n):
   total = 0
   while n > 0:
      digit = n % 10       # Get the last digit
      total=total+digit    # Add it to total
      n //= 10             # Remove the last digit
   return total

# Input from user
```

```
num = int(input("Enter a number: "))

# Make sure the number is positive
num = abs(num)

# Call the function and display result
result = sum_of_digits(num)
print(f"The sum of the digits of {num} is {result}")
```

#13. Write a program to check whether a given string is a palindrome or not.

```
def is_palindrome(s):
    # Remove spaces and convert to lowercase
    s = s.replace(" ", "").lower()
    # Compare string with its reverse
    return s == s[::-1]

# Input from user
text = input("Enter a string: ")

# Check and print result
if is_palindrome(text):
    print(f'"{text}" is a palindrome.')
else:
    print(f'"{text}" is not a palindrome.')
```

#14. Write a program to find the area of a rectangle.

```
def area_of_rectangle(length, width):
    return length * width

# Input from user
length = float(input("Enter the length of the rectangle: "))
width = float(input("Enter the width of the rectangle: "))

# Calculate and display area
area = area_of_rectangle(length, width)
print(f"The area of the rectangle is {area} square units.")
```

#15. Write a program to find the area of a circle.

```
def area_of_circle(radius):
    return math.pi * radius ** 2

# Input from user
radius = float(input("Enter the radius of the circle: "))
```

```
# Calculate and display area
area = area_of_circle(radius)
print(f"The area of the circle is {area:.2f} square units.")
```

#1.Write a program to find the second largest element in an array.

```
def find_second_largest(arr):
    if len(arr) < 2: # a list has must 2 elements
        return "Array must have at least two elements."
        unique_arr = list(set(arr))     # Remove duplicates
    if len(unique_arr) < 2:
        return "No second largest element (all elements are the same)."
    unique_arr.sort(reverse=True)  # Sort in descending order
    return unique_arr[1]

arr = [10, 20, 4, 45, 99, 99, 45]
second_largest = find_second_largest(arr)
print("Second largest element is:", second_largest)
```

#2.Write a program to sort a list of elements in ascending order.

```
def sort_list_ascending(arr):
    return sorted(arr) # sorted Works on any iterable (lists, tuples, dictionaries, etc.)

arr = [5, 2, 9, 1, 5, 6]
sorted_arr = sort_list_ascending(arr)
print("Sorted list in ascending order:", sorted_arr) # printing the list
```

#3. Write a program to remove duplicates from a list

```
def remove_duplicates_ordered(lst):
    seen = set()        # Create an empty set to track seen items
    result = []         # Create an empty list to store unique items

    for item in lst:    # Go through each element in the original list
        if item not in seen:   # If the item is not already seen
            seen.add(item)     # Add it to the seen set
            result.append(item)  # Also add it to the result list

    return result       #  Return the final list without duplicates
```

#4. Write a program to find the common elements between two lists

```
list1 = list(map(int, input("Enter the first list: ").split()))
list2 = list(map(int, input("Enter the second list: ").split()))

# Find common elements
common_elements = list(set(list1) & set(list2))

# Display result
print("Common elements:", common_elements)
```

#5. Write a program to find the factorial of a number using recursion.

```
def factorial(n):
    # Base case
    if n == 0 or n == 1:
        return 1
    # Recursive case
    else:
        return n * factorial(n - 1)

# Input from the user
num = int(input("Enter a number to find its factorial: "))

# Check if the input is valid
if num < 0:
    print("Factorial is not defined for negative numbers.")
else:
    result = factorial(num)
    print(f"The factorial of {num} is {result}")
```

#6. Write a program to find the LCM of two numbers.

```
import math
# Input from the user
a = int(input("Enter the first number: "))
b = int(input("Enter the second number: "))

# Calculate LCM using the formula
lcm = abs(a * b) // math.gcd(a, b)

# Display the result
print(f"The LCM of {a} and {b} is {lcm}")
```

#7.Write a program to implement binary search.

```
def binary_search(arr, target):
    low = 0
    high = len(arr) - 1

    while low <= high:
        mid = (low + high) // 2  # Find the middle index
        if arr[mid] == target:
            return mid  # Target found at index mid
        elif arr[mid] < target:
            low = mid + 1  # Search in the right half
        else:
            high = mid - 1  # Search in the left half

    return -1  # Target not found


# Input from the user
arr = list(map(int, input("Enter sorted list of numbers (space-separated): ").split()))
target = int(input("Enter the number to search for: "))

# Perform binary search
result = binary_search(arr, target)

# Output the result
if result != -1:
    print(f"Element found at index {result}")
else:
    print("Element not found in the list.")
```

#8. Write a program to implement selection sort.

```
def selection_sort(arr):
    n = len(arr)

    # Traverse through all elements in the array
    for i in range(n):
        # Assume the current index has the minimum value
        min_index = i

        # Find the index of the minimum element in the unsorted portion
        for j in range(i + 1, n):
            if arr[j] < arr[min_index]:
                min_index = j  # Update min_index if smaller value is found

        # Swap the found minimum element with the first unsorted element
        arr[i], arr[min_index] = arr[min_index], arr[i]

        # Print array after each pass (optional, for understanding)
        print(f"Step {i + 1}: {arr}")
```

```
# Input from the user
arr = list(map(int, input("Enter a list of numbers (space-separated): ").split()))

# Call the sorting function
selection_sort(arr)

# Output the sorted array
print("Sorted array:", arr)
```

#9.Write a program to implement merge sort.

```
def merge_sort(arr):
    # Base case: A list of 0 or 1 element is already sorted
    if len(arr) <= 1:
        return arr

    # Find the middle point to divide the array into two halves
    mid = len(arr) // 2

    # Recursively sort both halves
    left_half = merge_sort(arr[:mid])
    right_half = merge_sort(arr[mid:])

    # Merge the sorted halves
    return merge(left_half, right_half)

def merge(left, right):
    merged = []
    i = j = 0

    # Compare elements from both lists and add the smaller one
    while i < len(left) and j < len(right):
        if left[i] < right[j]:
            merged.append(left[i])
            i += 1
        else:
            merged.append(right[j])
            j += 1

    # Add any remaining elements from left or right
    merged.extend(left[i:])
    merged.extend(right[j:])
    return merged


# Input from the user
arr = list(map(int, input("Enter a list of numbers (space-separated): ").split()))
```

```
# Perform merge sort
sorted_arr = merge_sort(arr)

# Output the sorted array
print("Sorted array:", sorted_arr)
```

#10. Write a python program to implement quick sort

```
def quick_sort(array):
    # If the array has 0 or 1 elements, it's already sorted
    if len(array) <= 1:
        return array

    # Choose the last element as the pivot
    pivot = array[-1]

    # Divide elements into three lists based on comparison with the pivot
    smaller = []   # Elements less than pivot
    equal = []     # Elements equal to pivot
    greater = []   # Elements greater than pivot

    for element in array:
        if element < pivot:
            smaller.append(element)
        elif element == pivot:
            equal.append(element)
        else:
            greater.append(element)

    # Recursively sort the smaller and greater lists and combine them
    return quick_sort(smaller) + equal + quick_sort(greater)

numbers = [10, 7, 8, 9, 1, 5]
sorted_numbers = quick_sort(numbers)
print("Sorted list:", sorted_numbers)
```

#11.Write a python program to find the sum of all prime numbers between two numbers

```
def is_prime(number):
    # Numbers less than 2 are not prime
    if number < 2:
        return False

    # Check if the number is divisible by any number from 2 to sqrt(number)
    for i in range(2, int(number ** 0.5) + 1):
        if number % i == 0:
            return False  # Not a prime
    return True  # It's a prime
```

```
def sum_of_primes(start, end):
    total = 0  # To store the sum of prime numbers

    # Loop through each number in the given range
    for num in range(start, end + 1):
        if is_prime(num):  # Check if it's prime
            total += num   # Add it to the total sum

    return total

start = int(input("Enter the starting number: "))
end = int(input("Enter the ending number: "))

result = sum_of_primes(start, end)
print(f"The sum of prime numbers between {start} and {end} is: {result}")
```

#12.write a python program to find the sum of all the even numbers between two numbers

```
def sum_of_even_numbers(start, end):
    total = 0  # Variable to store the sum

    # Loop through the numbers from start to end
    for num in range(start, end + 1):
        if num % 2 == 0:  # Check if the number is even
            total += num  # Add it to the total sum

    return total

start = int(input("Enter the starting number: "))
end = int(input("Enter the ending number: "))

result = sum_of_even_numbers(start, end)
print(f"The sum of even numbers between {start} and {end} is: {result}")
```

#13. write a python program to find the sum of all the odd numbers between two odd given odd numbers

```
def sum_of_odd_numbers_between(start, end):
    total = 0

    if start > end:
        start, end = end, start
    current = start + 2

    # Loop until one less than 'end'
    while current < end:
        if current % 2 != 0:  # Check if current is odd (this is always true since we add +2)
```

```
            total += current
        current += 2

    return total

start = int(input("Enter the first odd number: "))
end = int(input("Enter the second odd number: "))

# Check if both numbers are odd
if start % 2 == 0 or end % 2 == 0:
    print("Please enter only odd numbers.")
else:
    result = sum_of_odd_numbers_between(start, end)
    print(f"The sum of odd numbers between {start} and {end} is: {result}")
```

#14.Write a python program to find th esruare root of a number using the Newton-Raphson Method

```
def newton_sqrt(number, tolerance=1e-10, max_iterations=1000):
    if number < 0:
        raise ValueError("Cannot compute square root of a negative number.")

    # Initial guess can be the number itself or number/2
    guess = number / 2 if number != 0 else 0

    for _ in range(max_iterations):
        # Apply Newton-Raphson formula
        next_guess = 0.5 * (guess + number / guess)

        # Check if the difference is within the desired tolerance
        if abs(next_guess - guess) < tolerance:
            return next_guess

        guess = next_guess

    return guess  # Return the last guess if max_iterations reached

num = float(input("Enter a non-negative number to find its square root: "))
result = newton_sqrt(num)
print(f"The square root of {num} is approximately {result}")
```

#15.Write a python program to find the power of a number using recursion

```
def power(base, exponent):
    # Base case: any number to the power 0 is 1
    if exponent == 0:
        return 1
    # If exponent is negative, compute the reciprocal
```

```
    elif exponent < 0:
        return 1 / power(base, -exponent)
    else:
        # Recursive case: multiply base by power(base, exponent - 1)
        return base * power(base, exponent - 1)

b = float(input("Enter the base number: "))
e = int(input("Enter the exponent (integer): "))

result = power(b, e)
print(f"{b} raised to the power {e} is {result}")
```