

## 1. Write a program that finds greatest of three numbers using functions. Pass the numbers as arguments.

```
In [6]: def find_greatest(a, b, c):
        """
        Find the greatest number among three given numbers.

        Args:
        a (float): First number
        b (float): Second number
        c (float): Third number

        Returns:
        float: The greatest number among a, b, and c
        """
        return max(a, b, c)

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

#Find and display the greatest number
result = find_greatest(num1, num2, num3)
print(f"The greatest number among {num1}, {num2}, and {num3} is:-> {result}")

Enter the first number: 54
Enter the second number: 24
Enter the third number: 2
The greatest number among 54.0, 24.0, and 2.0 is:-> 54.0
```

## 2. Write a program to implement these formulae of permutations and combinations.

Number of permutations of n objects taken r at a time:  $p(n, r) = n! / (n-r)!$ .

Number of combinations of n objects taken r at a time is:  $c(n, r) = n! / (r!(n-r)!)$   
 $= p(n, r) / r!$

```
In [7]: def factorial(n):
        """Calculate the factorial of a number."""
        if n == 0 or n == 1:
            return 1
        else:
            return n * factorial(n - 1)
```

```

def permutation(n, r):
    """
    Calculate the number of permutations of n objects taken r at a time.
    Formula:  $P(n,r) = n! / (n-r)!$ 
    """
    return factorial(n) // factorial(n - r)

def combination(n, r):
    """
    Calculate the number of combinations of n objects taken r at a time.
    Formula:  $C(n,r) = n! / (r! * (n-r)!) = P(n,r) / r!$ 
    """
    return permutation(n, r) // factorial(r)

#Input from the user
n = int(input("Enter the total number of objects (n): "))
r = int(input("Enter the number of objects to be chosen (r): "))

#Calculate and display results
print(f"Number of permutations P({n},{r}): {permutation(n, r)}")
print(f"Number of combinations C({n},{r}): {combination(n, r)}")

Enter the total number of objects (n): 5
Enter the number of objects to be chosen (r): 4
Number of permutations P(5,4): 120
Number of combinations C(5,4): 5

```

3. Write a function `cubesum()` that accepts an integer and returns the sum of the cubes of individual digits of that number. Use this function to make functions `PrintArmstrong()` and `isArmstrong()` to print Armstrong numbers and to find whether is an Armstrong number.

```

In [9]: def cubesum(num):
    """
    Calculate the sum of cubes of individual digits of a number.

    Args:
    num (int): The input number

    Returns:
    int: Sum of cubes of individual digits
    """
    return sum(int(digit)**3 for digit in str(num))

def isArmstrong(num):
    """
    Check if a number is an Armstrong number.

    Args:
    num (int): The number to check

    Returns:
    bool: True if the number is an Armstrong number, False otherwise
    """

```

```

    """
    return num == cubesum(num)

def PrintArmstrong(start, end):
    """
    Print all Armstrong numbers in a given range.

    Args:
    start (int): Start of the range (inclusive)
    end (int): End of the range (inclusive)
    """
    armstrong_numbers = [num for num in range(start, end+1) if isArmstrong(num)]
    if armstrong_numbers:
        print(f"Armstrong numbers between {start} and {end} are:")
        print(", ".join(map(str, armstrong_numbers)))
    else:
        print(f"There are no Armstrong numbers between {start} and {end}.")

# Example usage
print("User Given input of cubesum():")
num = 153
print(f"The sum of cubes of digits of {num} is: {cubesum(num)}")

print("\nUser Given input of isArmstrong():")
print(f"Is {num} an Armstrong number? {isArmstrong(num)}")

print("\nUser Given input of PrintArmstrong():")
PrintArmstrong(100, 1000)

# Interactive part
print("\nLet's check a number of your choice:")
user_num = int(input("Enter a number to check if it's an Armstrong number: "))
if isArmstrong(user_num):
    print(f"{user_num} is an Armstrong number!")
else:
    print(f"{user_num} is not an Armstrong number.")

```

User Given input of cubesum():  
The sum of cubes of digits of 153 is: 153

User Given input of isArmstrong():  
Is 153 an Armstrong number? True

User Given input of PrintArmstrong():  
Armstrong numbers between 100 and 1000 are:  
153, 370, 371, 407

Let's check a number of your choice:  
Enter a number to check if it's an Armstrong number: 153  
153 is an Armstrong number!

## 4. Write a Python function to create and print a list where the values are the squares of numbers between 1 and 30 (both included).

```

In [10]: def create_and_print_squares():
    """
    Create and print a list of squares for numbers between 1 and 30 (inclusive).
    """

```

```
squares_list = [num**2 for num in range(1, 31)]
print("List of squares from 1 to 30:")
print(squares_list)
```

```
#Call the function to execute it
create_and_print_squares()
```

List of squares from 1 to 30:

[1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 361, 400, 441, 484, 529, 576, 625, 676, 729, 784, 841, 900]

**5. Given a string  $s = "1234"$  and an integer  $n = 5678$ , concatenate them as a single string and then convert the result back to an integer. What is the final integer value?**

```
In [11]: s = "1234"
n = 5678

#Step 1: Convert integer n to a string
n_str = str(n)

#Step 2: Concatenate the string s and the string version of n
concatenated_string = s + n_str

#Step 3: Convert the concatenated string back to an integer
final_integer = int(concatenated_string)

#Print the final integer value
print(final_integer)
```

12345678

**6. Write a Python program that repeatedly asks the user to enter a positive integer. If the user enters a negative number or zero, the program should ask again until a positive integer is entered.**

```
In [12]: def get_positive_integer():
num = -1 #Initialize with a non-positive value
while num <= 0:
    user_input = input("Please enter a positive integer: ")

    if user_input.isdigit(): #Check if the input is a positive number
        num = int(user_input) #Convert it to an integer
        if num > 0:
            print(f"Thank you! You entered: {num}")
        else:
            print("The number must be greater than zero. Try again.")
    else:
        print("Invalid input. Please enter a positive integer.")

#Call the function
get_positive_integer()
```

```
Please enter a positive integer: 5  
Thank you! You entered: 5
```

In [ ]: