```
#This notebook contains various examples related to following concepts
### Strings, arrays , matrices and its operations and strings with it's
operations
#### The notebook can be viewed at (https://github.com/amitvsuryavanshi04/
SIC_programming_and_coding)
```

This notebook contains various examples related to following concepts

Strings, arrays , matrices and its operations and strings with it's operations

The notebook can be viewed at (https://github.com/amitvsuryavanshi04/SIC_programming_and_coding)

Program 31 code to get cube sum of first n natural numbers ex 1^3 + 2^3 +3^3 = 1+8+27 = 36

```
def cube_sum_of_natural_numbers(n):
  if n <= 0:
    return 0
  else:
    total = sum([i**3 for i in range(1,n+1)])
    return total

#input the number of natural numbers
n = int(input ("Enter the value of n: "))

if n <= 0:
  print("Please enter a positive integer.")
else:
  result = cube_sum_of_natural_numbers(n)
  print(f"The cube sum of the first {n} natural numbers is: {result}")
```

```
Enter the value of n: 3
The cube sum of the first 3 natural numbers is: 36
```

Program 32 to find sum of array.

```
# Finding Sum of Array Using sum()
arr = [1,2,3]
ans = sum(arr)
print('Sum of the array is ', ans)
```

```
Sum of the array is  6
```

```
# Function to find the sum of elements in an array
def sum_of_array(arr):
  total = 0 # Initialize a variable to store the sum
  for element in arr:
    total += element # Add each element to the total
  return total
# Example usage:
array = [1, 2, 3]
result = sum_of_array(array)
print("Sum of the array:", result)
```

```
Sum of the array: 6
```

Program 33 to find the largest element in an array.

```
def find_largest_element(arr):
  if not arr:
    return "Array is empty"
# Initialize the first element as the largest
  largest_element = arr[0]
# Iterate through the array to find the largest element
  for element in arr:
    if element > largest_element:
      largest_element = element
  return largest_element
# Example usage:
my_array = [10, 20, 30, 99]
result = find_largest_element(my_array)
print(f"The largest element in the array is: {result}")
```

```
The largest element in the array is: 99
```

Program 34 for array rotation

```
def rotate_array(arr, d):
    n = len(arr)
    # Check if 'd' is valid, it should be within the range of array len
    if d < 0 or d >= n:
        return "Invalid rotation value"

    # Create a new array to store the rotated elements.
    rotated_arr = [0] * n

    # Perform the rotation.
    for i in range(n):
        rotated_arr[i] = arr[(i + d) % n]

    return rotated_arr

# Input array
arr = [1, 2, 3, 4, 5]
# Number of positions to rotate
d = 3
# Call the rotate_array function
result = rotate_array(arr, d)
# Print the rotated array
print("Original Array:", arr)
print("Rotated Array:", result)
```

```
Original Array: [1, 2, 3, 4, 5]
Rotated Array: [4, 5, 1, 2, 3]
```

Program 35 split the array and add the first part to the end ?

```
def split_and_add(arr, k):
  if k <= 0 or k >= len(arr):
    return arr
# Split the array into two parts
  first_part = arr[:k]
  second_part = arr[k:]
# Add the first part to the end of the second part
```

```
  result = second_part + first_part
  return result
# Test the function
arr = [1, 2, 3, 4, 5]
k = 3
result = split_and_add(arr, k)
print("Original Array:", arr)
print("Array after splitting and adding:", result)
```

```
Original Array: [1, 2, 3, 4, 5]
Array after splitting and adding: [4, 5, 1, 2, 3]
```

Program 36 to check if given array is monotonic or not ? A monotonic array is that which is entirely either non-increasing , or non-decreasing

```
def is_monotonic(arr):
  increasing = decreasing = True
  for i in range(1, len(arr)):
    if arr[i] > arr[i - 1]:
      decreasing = False
    elif arr[i] < arr[i - 1]:
      increasing = False
  return increasing or decreasing
# Test the function
arr1 = [1, 2, 2, 3] # Monotonic (non-decreasing)
arr2 = [3, 2, 1] # Monotonic (non-increasing)
arr3 = [1, 3, 2, 4] # Not monotonic
print("arr1 is monotonic:", is_monotonic(arr1))
print("arr2 is monotonic:", is_monotonic(arr2))
print("arr3 is monotonic:", is_monotonic(arr3))
```

```
arr1 is monotonic: True
arr2 is monotonic: True
arr3 is monotonic: False
```

Program 37 to add two matrices

```
# Function to add two matrices
def add_matrices(mat1, mat2):
# Check if the matrices have the same dimensions
  if len(mat1) != len(mat2) or len(mat1[0]) != len(mat2[0]):
    return "Matrices must have the same dimensions for addition"
```

```
# Initialize an empty result matrix with the same dimensions
  result = []
  for i in range(len(mat1)):
    row = []
    for j in range(len(mat1[0])):
      row.append(mat1[i][j] + mat2[i][j])
    result.append(row)
  return result
# Input matrices
matrix1 = [
[1, 2, 3],
[4, 5, 6],
[7, 8, 9]
]
matrix2 = [
[9, 8, 7],
[6, 5, 4],
[3, 2, 1]
]
# Call the add_matrices function
result_matrix = add_matrices(matrix1, matrix2)
# Display the result
if isinstance(result_matrix, str):
  print(result_matrix)
else:
  print("Sum of matrices:")
for row in result_matrix:
  print(row)
```

```
Sum of matrices:
[10, 10, 10]
[10, 10, 10]
[10, 10, 10]
```

Program 38 to multiply two matrices

```
def multiply_matrices(mat1, mat2):
# Determine the dimensions of the input matrices
  rows1 = len(mat1)
  cols1 = len(mat1[0])
  rows2 = len(mat2)
  cols2 = len(mat2[0])
```

```
# Check if multiplication is possible
  if cols1 != rows2:
    return "Matrix multiplication is not possible. Number of column"
# Initialize the result matrix with zeros
  result = [[0 for _ in range(cols2)] for _ in range(rows1)]
# Perform matrix multiplication
  for i in range(rows1):
    for j in range(cols2):
      for k in range(cols1):
        result[i][j] += mat1[i][k] * mat2[k][j]
  return result
# Example matrices
matrix1 = [[1, 2, 3],
           [4, 5, 6]]
matrix2 = [[7, 8],
           [9, 10],
           [11, 12]]
# Multiply the matrices
result_matrix = multiply_matrices(matrix1, matrix2)
# Display the result
if isinstance(result_matrix, str):
  print(result_matrix)
else:
  print("Result of matrix multiplication:")
for row in result_matrix:
  print(row)
```

```
Result of matrix multiplication:
[58, 64]
[139, 154]
```

Program 39 code for transpose of a matrix

```
# Function to transpose a matrix
def transpose_matrix(matrix):
  rows, cols = len(matrix), len(matrix[0])
# Create an empty matrix to store the transposed data
  result = [[0 for _ in range(rows)] for _ in range(cols)]
  for i in range(rows):
    for j in range(cols):
      result[j][i] = matrix[i][j]
  return result
```

```
# Input matrix
matrix = [
[1, 2, 3],
[4, 5, 6]
]
# Transpose the matrix
transposed_matrix = transpose_matrix(matrix)
# Print the transposed matrix
for row in transposed_matrix:
  print(row)
```

```
[1, 4]
[2, 5]
[3, 6]
```

Program 40 program to sort words in alphabetic order

```
# Program to sort alphabetically the words form a string provided by th
my_str = input("Enter a string: ")
# breakdown the string into a list of words
words = [word.capitalize() for word in my_str.split()]
# sort the list
words.sort()
# display the sorted words
print("The sorted words are:")
for word in words:
  print(word)
```

```
Enter a string: amit vthathrathrayan pavan sudarshan ekta pratheeksha chinmai
The sorted words are:
Amit
Chinmai
Ekta
Pavan
Pratheeksha
Sudarshan
Vthathrathrayan
```