
Write a program to create an array [10, 5, 7, 20, 15, -9, 33], print it, Find the maximum and minimum Value of the array

```
#Code here
arr = [10, 5, 7, 20, 15, -9, 33]
print("Array:", arr)
print("Maximum Value:", max(arr))
print("Minimum Value:", min(arr))
```

```
Array: [10, 5, 7, 20, 15, -9, 33]
Maximum Value: 33
Minimum Value: -9
```

Write a program to accept an array of elements and find the maximum without using max function

```
#Code here
elements = list(map(int, input("Enter elements separated by space:
").split()))
max_value = elements[0]
for num in elements:
    if num > max_value:
        max_value = num
print("Maximum Value:", max_value)
```

```
Enter elements separated by space:  4 6 8 5 9

Maximum Value: 9
```

Write a program to accept two matrices [[1,2],[3,4]], [[2,4],[6,8]]and find the sum and subtraction of two matrices

```
#Code here
matrix1 = [[1, 2], [3, 4]]
matrix2 = [[2,
4], [6, 8]]

def add_matrices(mat1, mat2):
    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

def subtract_matrices(mat1, mat2):
    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

sum_matrix = add_matrices(matrix1, matrix2)
subtraction_matrix = subtract_matrices(matrix1, matrix2)

print("Sum of Matrices:")
for row in sum_matrix:
    print(row)

print("\nSubtraction of Matrices:")
for row in subtraction_matrix:
    print(row)
```

```
Sum of Matrices:
[3, 6]
[9, 12]

Subtraction of Matrices:
[-1, -2]
[-3, -4]
```

Write a program to generate a ranodom matrix of size 5 X 6. Add another row containing random elements to it. Find the sum of each row.

```
#Code here
matrix = np.random.randint(1, 100, size=(5,
6))
new_row = np.random.randint(1, 100, size=(1,
6))
matrix = np.vstack([matrix, new_row])
row_sums = np.sum(matrix, axis=1)
print("Matrix:")
print(matrix)
print("\nSum of Each Row:")
print(row_sums)
```

```
Matrix:
[[77 21  1 52 58 58]
 [42  9 99 58 75 10]
```

```
 [85 37  1 69 33 60]
 [69 60 18 47 22 81]
 [11 76 66 33  1 53]
 [44 33 95 28 39 66]]

Sum of Each Row:
[267 293 285 297 240 305]
```

Write a program to generate a ranodm integer matrix of size 5 X 6. Add another column containing random integers to it. Compute the sine of each element and print it

```
#Code here
matrix = np.random.randint(1, 100, size=(5, 6))
new_column = np.random.randint(1, 100, size=(5,
1))
matrix = np.hstack([matrix, new_column])
sine_matrix = np.sin(matrix)
print("Matrix:")
print(matrix)
print("\nSine of Each Element:")
print(sine_matrix)

Matrix:
[[40 15 56 17 45 53 83]
 [27 80 33 98 12 98 26]
 [95 11 29 25  5 36 31]
 [27 53 93 50 94  4  9]
 [73 45 40 55 25 48 44]]

Sine of Each Element:
[[ 0.74511316  0.65028784 -0.521551   -0.96139749  0.85090352
0.39592515
   0.96836446]
 [ 0.95637593 -0.99388865  0.99991186 -0.57338187 -0.53657292 -
0.57338187
   0.76255845]
 [ 0.68326171 -0.99999021 -0.66363388 -0.13235175 -0.95892427 -
0.99177885
  -0.40403765]
 [ 0.95637593  0.39592515 -0.94828214 -0.26237485 -0.24525199 -
0.7568025
   0.41211849]
 [-0.67677196  0.85090352  0.74511316 -0.99975517 -0.13235175 -
0.76825466
   0.01770193]]
```

Write a program to generate a ranodm matrix of size 5 X 6. Consider the 3rd and 4th row; and 2nd, 3rd and 4th column and print the resultant matrix using slicing

```
#Code here
matrix = np.random.randint(1, 100, size=(5, 6))
resultant_matrix = matrix[2:4,
1:4]
print("Original Matrix:")
print(matrix)
print("\nResultant Matrix (Rows 3-4, Columns 2-4):")
print(resultant_matrix)
```

```
Original Matrix:
[[47 76 64 27 33 74]
 [ 6 67 12 99 16 80]
 [44 67  6 88 73 11]
 [38 65 20 62 72 81]
 [57 77 83 88 65 18]]

Resultant Matrix (Rows 3-4, Columns 2-4):
[[67  6 88]
 [65 20 62]]
```

Great Job!