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

In [1]:

```
#Code here
import numpy as np

arr = np.array([10, 5, 7, 20, 15, -9, 33])
print("Array:", arr)
print("Maximum Value:", np.max(arr))
print("Minimum Value:", np.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

In [3]:

```
#Code here
arr2 = list(map(int, input("Enter array elements separated by space: ").split()))
max_val = arr2[0]
for num in arr2:
    if num > max_val:
        max_val = num
print("Maximum value without using max():", max_val)
```

```
Maximum value without using max(): 34
```

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

In [4]:

```
#Code here
A = np.array([[1, 2], [3, 4]])
B = np.array([[2, 4], [6, 8]])

sum_matrix = A + B
diff_matrix = A - B

print("Matrix A:\n", A)
print("Matrix B:\n", B)
print("Sum of Matrices:\n", sum_matrix)
print("Subtraction of Matrices:\n", diff_matrix)
```

```
Matrix A:
 [[1 2]
 [3 4]]
Matrix B:
 [[2 4]
 [6 8]]
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.

In [5]:

```
#Code here
matrix = np.random.randint(1, 50, (5, 6))
new_row = np.random.randint(1, 50, (1, 6))
updated_matrix = np.vstack((matrix, new_row))
row_sums = np.sum(updated_matrix, axis=1)

print("Original Matrix:\n", matrix)
print("New Row:\n", new_row)
print("Updated Matrix:\n", updated_matrix)
print("Sum of Each Row:", row_sums)
```

```
Original Matrix:
 [[11 24  8 10 19 32]
 [39 36 45 16 21 36]
 [33 23  4 46 12 19]
 [14  9  8  4 36 22]
 [26 42 49 10  9 10]]
New Row:
 [[15  8 42 31 12 49]]
Updated Matrix:
 [[11 24  8 10 19 32]
 [39 36 45 16 21 36]
 [33 23  4 46 12 19]
 [14  9  8  4 36 22]
 [26 42 49 10  9 10]
 [15  8 42 31 12 49]]
Sum of Each Row: [104 193 137  93 146 157]
```

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

In [6]:

```
#Code here
matrix = np.random.randint(1, 50, (5, 6))
new_col = np.random.randint(1, 50, (5, 1))
```

```
updated_matrix = np.hstack((matrix, new_col))
sine_matrix = np.sin(updated_matrix)

print("Original Matrix:\n", matrix)
print("New Column:\n", new_col)
print("Updated Matrix:\n", updated_matrix)
print("Sine of Each Element:\n", sine_matrix)
```

```
Original Matrix:
 [[28  5 46  5 10 29]
 [19 28 17 20  3 31]
 [21 36 20 32 44 21]
 [31 31 20 38 16  9]
 [35 29 37 24  8 29]]
New Column:
 [[20]
 [35]
 [38]
 [ 4]
 [36]]
Updated Matrix:
 [[28  5 46  5 10 29 20]
 [19 28 17 20  3 31 35]
 [21 36 20 32 44 21 38]
 [31 31 20 38 16  9  4]
 [35 29 37 24  8 29 36]]
Sine of Each Element:
 [[ 0.27090579 -0.95892427  0.90178835 -0.95892427 -0.54402111 -0.66363388
   0.91294525]
 [ 0.14987721  0.27090579 -0.96139749  0.91294525  0.14112001 -0.40403765
  -0.42818267]
 [ 0.83665564 -0.99177885  0.91294525  0.55142668  0.01770193  0.83665564
   0.29636858]
 [-0.40403765 -0.40403765  0.91294525  0.29636858 -0.28790332  0.41211849
  -0.7568025 ]
 [-0.42818267 -0.66363388 -0.64353813 -0.90557836  0.98935825 -0.66363388
  -0.99177885]]
```

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

In [7]:

```
#Code here
matrix = np.random.randint(1, 100, (5, 6))
sliced_matrix = matrix[2:4, 1:4]

print("Original Matrix:\n", matrix)
print("Sliced Matrix (3rd & 4th row, 2nd to 4th col):\n", sliced_matrix)
```

```
Original Matrix:
 [[ 6 45 99 26 55 71]
 [53 35 90 90  2 78]
 [59 71 19 71 61  3]
 [89 40 40 34 82 96]
 [72 28 83 48 29 49]]
Sliced Matrix (3rd & 4th row, 2nd to 4th col):
 [[71 19 71]
 [40 40 34]]
```

Great Job!