```python
# Question 1
import numpy as np
print("NumPy Version:", np.__version__)

# Taking input list from user
user_input = input("Enter elements separated by space: ").split()

freq = {}
for x in user_input:
    freq[x] = freq.get(x, 0) + 1

print("Frequency of elements:", freq)
```

```
NumPy Version: 2.0.1

Enter elements separated by space:  1 2 3 4 5

Frequency of elements: {'1': 1, '2': 1, '3': 1, '4': 1, '5': 1}
```

```python
#Question 2
s = input("Enter a binary string: ")

ones = s.count('1')
zeros = s.count('0')

print("Output:", '1' * ones + '0' * zeros)
```

```
Enter a binary string:  1110011101

Output: 1111111000
```

```python
#Question 3
s = input("Enter a string: ")
n = int(input("Enter index to remove: "))

new_s = s[:n] + s[n+1:]
print("Result:", new_s)
```

```
Enter a string:  Open Source Software Lab
Enter index to remove:  8

Result: Open Souce Software Lab
```

```python
# Question 4
import numpy as np

arr = np.ones((5,5))
arr[1:-1, 1:-1] = 0
print(arr)
```

```
[[1. 1. 1. 1. 1.]
 [1. 0. 0. 0. 1.]
 [1. 0. 0. 0. 1.]
 [1. 0. 0. 0. 1.]
 [1. 1. 1. 1. 1.]]
```

```
# Question 5
import numpy as np

array1 = np.array([0, 10, 20, 40, 60])
array2 = np.array([0, 40])

result = np.in1d(array1, array2)
print(result)
```

```
[ True False False  True False]
```

```
/tmp/ipykernel_505/3291685827.py:6: DeprecationWarning: `in1d` is
deprecated. Use `np.isin` instead.
  result = np.in1d(array1, array2)
```

```
# Question 6
import numpy as np

array1 = np.array([0, 10, 20, 40, 60, 80])
array2 = np.array([10, 30, 40, 50, 70])

result = np.setxor1d(array1, array2)
print(result)
```

```
[ 0 20 30 50 60 70 80]
```

```
# Question 7
import numpy as np

a = np.array([1, 2, 3])
b = np.array([4, 5, 6])

result = np.column_stack((a, b))
print(result)
```

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

```
# Question 8
import numpy as np

n = int(input("Enter size of square matrix: "))

print("Enter rows:")
mat = []
for _ in range(n):
    row = list(map(float, input().split()))
    mat.append(row)

mat = np.array(mat)

print("Matrix:\n", mat)
print("Rank:", np.linalg.matrix_rank(mat))
print("Trace:", np.trace(mat))
print("Determinant:", np.linalg.det(mat))
```

```
Enter size of square matrix:  2

Enter rows:

 1 2
 2 1

Matrix:
 [[1. 2.]
 [2. 1.]]
Rank: 2
Trace: 2.0
Determinant: -2.9999999999999996
```