```
# Employee Salary Data Analysis
import numpy as np

salaries = [25000, 30000, 28000, 32000, 29000, 31000, 27000, 35000, 26000]

# 1. Create 1D array
arr1 = np.array(salaries)
print("1D Array:", arr1)

# 2. Convert to 2D (3x3)
arr2 = arr1.reshape(3,3)
print("2D Array:\n", arr2)

# 3. Shape, dtype, element access
print("Shape:", arr2.shape)
print("Data Type:", arr2.dtype)
print("Salary at row 2 column 1:", arr2[1,0])

# 4. Slicing
print("Index 2 to 6:", arr1[2:7])
print("Last 3 salaries:", arr1[-3:])

# 5. Sorting
print("Ascending:", np.sort(arr1))
print("Descending:", np.sort(arr1)[::-1])

# 6. Reshape back to 1D
arr_back = arr2.reshape(-1)
print("Reshaped to 1D:", arr_back)

# 7. Join with bonus
bonus = np.array([2000, 3000, 2500, 4000, 1500, 3500, 2800, 5000, 1800])
joined = np.concatenate((arr1, bonus))
print("Joined Array:", joined)
```

 # Product Stock Management

```
import numpy as np

stock = np.array([45, 60, 30, 80, 55, 90, 20, 70])

# 1. zeros, ones, arange
print("Zeros:", np.zeros(8))
print("Ones:", np.ones(8))
print("Arange 10-50 step 5:", np.arange(10, 50, 5))

# 2. Convert to 2D (4x2)
stock2d = stock.reshape(4,2)
print("2D Stock:\n", stock2d)

# 3. Access elements
print("Row 3 Column 1:", stock2d[2,0])
print("First Row:", stock2d[0])

# 4. Slicing
print("Index 1 to 5:", stock[1:6])
print("Index -4 to -1:", stock[-4:-1])

# 5. Searching
print("Index where stock is 90:", np.where(stock == 90))
print("Values > 50:", stock[stock > 50])

# 6. Splitting
print("Split into 4 parts:", np.split(stock,4))
print("Vertical Split:", np.vsplit(stock2d,2))
print("Horizontal Split:", np.hsplit(stock2d,2))
```

```
# Temperature Monitoring System
import numpy as np

temperature = np.array([30, 32, 31, 29, 35, 36, 33, 34, 28, 27, 26, 25])

# 1. 1D array
print("1D Array:", temperature)

# 2. Convert to 3D (2x2x3)
temp3d = temperature.reshape(2,2,3)
print("3D Array:\n", temp3d)

# 3. Access element
print("First block, second row, third column:",
      temp3d[0,1,2])

# 4. Data type
print("Data Type:", temperature.dtype)
temperature_float = temperature.astype(float)
print("Changed to Float:", temperature_float.dtype)

# 5. Slicing
print("Index 3 to 8:", temperature[3:9])
print("Every second value:", temperature[::2])

# 6. Sorting
print("Ascending:", np.sort(temperature))
print("Descending:", np.sort(temperature)[::-1])

# 7. Reshape into 4x3
temp4x3 = temperature.reshape(4,3)
print("Reshaped 4x3:\n", temp4x3)
```

```
# Student Roll Number Processing
import numpy as np

roll_numbers = np.arange(101,111)
print("Roll Numbers:", roll_numbers)

# Zeros & Ones
print("Zeros:", np.zeros(10))
print("Ones:", np.ones(10))

# Convert to 2D (5x2)
roll2d = roll_numbers.reshape(5,2)
print("2D Roll Numbers:\n", roll2d)

# Join with extra roll numbers
extra_roll = np.array([111,112,113,114,115])
joined_roll = np.concatenate((roll_numbers, extra_roll))
print("Joined Roll Numbers:", joined_roll)

# Search
print("Index of 105:", np.where(joined_roll == 105))
print("Roll numbers > 107:", joined_roll[joined_roll > 107])

# Split into 3 equal parts
split_roll = np.split(joined_roll,3)
print("Split into 3 parts:", split_roll)

# Shape check
print("Shape before reshape:", joined_roll.shape)
reshaped = joined_roll.reshape(3,5)
print("Shape after reshape:", reshaped.shape)
```

Outputs:

```
PS D:\Internship\Day15> python task1.py
1D Array: [25000 30000 28000 32000 29000 31000 27000 35000 26000]
2D Array:
 [[25000 30000 28000]
 [32000 29000 31000]
 [27000 35000 26000]]
Shape: (3, 3)
Data Type: int64
Salary at row 2 column 1: 32000
Index 2 to 6: [28000 32000 29000 31000 27000]
Last 3 salaries: [27000 35000 26000]
Ascending: [25000 26000 27000 28000 29000 30000 31000 32000 35000]
Descending: [35000 32000 31000 30000 29000 28000 27000 26000 25000]
Reshaped to 1D: [25000 30000 28000 32000 29000 31000 27000 35000 26000]
Joined Array: [25000 30000 28000 32000 29000 31000 27000 35000 26000  2000  3000  2500
  4000  1500  3500  2800  5000  1800]
```

```
PS D:\Internship\Day15> python task2.py
Zeros: [0. 0. 0. 0. 0. 0. 0. 0.]
Ones: [1. 1. 1. 1. 1. 1. 1. 1.]
Arange 10-50 step 5: [10 15 20 25 30 35 40 45]
2D Stock:
 [[45 60]
 [30 80]
 [55 90]
 [20 70]]
Row 3 Column 1: 55
First Row: [45 60]
Index 1 to 5: [60 30 80 55 90]
Index -4 to -1: [55 90 20]
Index where stock is 90: (array([5]),)
Values > 50: [60 80 55 90 70]
Split into 4 parts: [array([45, 60]), array([30, 80]), array([55, 90]), array([20, 70])]
Vertical Split: [array([[45, 60],
       [30, 80]]), array([[55, 90],
       [20, 70]])]
Horizontal Split: [array([[45],
       [30],
       [55],
       [20]]), array([[60],
       [80],
       [90],
       [70]])]
```

```
PS D:\Internship\Day15> python task3.py
1D Array: [30 32 31 29 35 36 33 34 28 27 26 25]
3D Array:
 [[[30 32 31]
  [29 35 36]]

 [[33 34 28]
  [27 26 25]]]
First block, second row, third column: 36
Data Type: int64
Changed to Float: float64
Index 3 to 8: [29 35 36 33 34 28]
Every second value: [30 31 35 33 28 26]
Ascending: [25 26 27 28 29 30 31 32 33 34 35 36]
Descending: [36 35 34 33 32 31 30 29 28 27 26 25]
Reshaped 4x3:
 [[30 32 31]
 [29 35 36]
 [33 34 28]
 [27 26 25]]
```

```
PS D:\Internship\Day15> python task4.py
Roll Numbers: [101 102 103 104 105 106 107 108 109 110]
Zeros: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
Ones: [1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
2D Roll Numbers:
 [[101 102]
 [103 104]
 [105 106]
 [107 108]
 [109 110]]
Joined Roll Numbers: [101 102 103 104 105 106 107 108 109 110 111 112 113 114 115]
Index of 105: (array([4]),)
Roll numbers > 107: [108 109 110 111 112 113 114 115]
Split into 3 parts: [array([101, 102, 103, 104, 105]), array([106, 107, 108, 109, 110]), array([111, 112, 113, 114, 115])]
Shape before reshape: (15,)
Shape after reshape: (3, 5)
```