In [ ]:
```python
# Write a program to distinguish between Array Indexing and Fancy Indexing.
import numpy as np

# NumPy array
arr = np.array([0, 1, 2, 3, 4, 5])

# Array Indexing
index_1 = arr[2]  # Access the element at index 2
index_2 = arr[1:4]  # Access elements from index 1(inclusive) to 3 (exclusive)
print("Array Indexing:")
print("Element at index 2:", index_1)
print("Elements from index 1 to 3 (exclusive):", index_2)

# Fancy Indexing
index_array = np.array([0, 4, 5])  # Array of integers
boolean_array = np.array([True, False, True, False, False, True])  # Boolean array

fancy_1 = arr[index_array]  # Access elements at specified indices using an integer
fancy_2 = arr[boolean_array]  # Access elements based on a boolean condition
print("\nFancy Indexing:")
print("Elements at specified indices:", fancy_1)
print("Elements based on a boolean condition:", fancy_2)
```

```
Array Indexing:
Element at index 2: 2
Elements from index 1 to 3 (exclusive): [1 2 3]

Fancy Indexing:
Elements at specified indices: [0 4 5]
Elements based on a boolean condition: [0 2 5]
```

In [ ]:
```python
# Execute the 2D array Slicing.

import numpy as np

# Create a sample 2D NumPy array
arr_2d = np.array([[1, 2, 3],
                   [4, 5, 6],
                   [7, 8, 9]])

# Slicing the 2D array
subarray_1 = arr_2d[0:2, 1:3]  # Rows 0 to 1 (exclusive) and Columns 1 to 2 (exclus
subarray_2 = arr_2d[:, 1]  # All rows and Column 1
subarray_3 = arr_2d[1, :]  # Row 1 and All columns

print("Original 2D Array:")
print(arr_2d)
print("\nSliced Subarrays:")
print("Subarray 1:")
print(subarray_1)
print("Subarray 2:")
print(subarray_2)
print("Subarray 3:")
print(subarray_3)
```

```
Original 2D Array:
[[1 2 3]
 [4 5 6]
 [7 8 9]]

Sliced Subarrays:
Subarray 1:
[[2 3]
 [5 6]]
Subarray 2:
[2 5 8]
Subarray 3:
[4 5 6]
```

In [ ]:
```python
# Create the 5-Dimensional arrays using ‘ndmin’.
import numpy as np

# Create a 5-dimensional array with ndmin
arr_5d = np.array([1, 2, 3], ndmin=5)

# Check the shape of the 5-dimensional array
print("Shape of the 5-dimensional array:", arr_5d.shape)

# Print the 5-dimensional array
print("5-dimensional array:")
print(arr_5d)
```

```
Shape of the 5-dimensional array: (1, 1, 1, 1, 3)
5-dimensional array:
[[[[[1 2 3]]]]]
```

In [ ]:
```python
# Reshape the array from 1-D to 2-D array.
import numpy as np

# Create a 1-D array
arr_1d = np.array([1, 2, 3, 4, 5, 6])

# Reshape the 1-D array to a 2-D array
arr_2d = arr_1d.reshape((2, 3))  # Specify the desired shape (2 rows, 3 columns)

# Alternatively, you can use np.reshape() function:
# arr_2d = np.reshape(arr_1d, (2, 3))

# Print the original and reshaped arrays
print("Original 1-D array:")
print(arr_1d)

print("\nReshaped 2-D array:")
print(arr_2d)
```

```
Original 1-D array:
[1 2 3 4 5 6]

Reshaped 2-D array:
[[1 2 3]
 [4 5 6]]
```

In [ ]:
```python
# Perform the Stack functions in Numpy arrays - Stack(), hstack(), vstack(), and ds
import numpy as np

# Create sample arrays
arr1 = np.array([1, 2, 3])
arr2 = np.array([4, 5, 6])

# np.stack(): Stacking along a new axis
stacked_axis0 = np.stack((arr1, arr2), axis=0)
stacked_axis1 = np.stack((arr1, arr2), axis=1)

print("np.stack() along axis 0:")
print(stacked_axis0)
print("\nnp.stack() along axis 1:")
print(stacked_axis1)

# np.hstack(): Stacking horizontally
hstacked = np.hstack((arr1, arr2))
print("\nnp.hstack():")
print(hstacked)

# np.vstack(): Stacking vertically
vstacked = np.vstack((arr1, arr2))
print("\nnp.vstack():")
print(vstacked)

# Create 2D arrays
arr3 = np.array([[7], [8], [9]])
arr4 = np.array([[10], [11], [12]])

# np.dstack(): Stacking along the third axis (depth-wise)
dstacked = np.dstack((arr3, arr4))
print("\nnp.dstack():")
print(dstacked)
```

```
np.stack() along axis 0:
[[1 2 3]
 [4 5 6]]

np.stack() along axis 1:
[[1 4]
 [2 5]
 [3 6]]

np.hstack():
[1 2 3 4 5 6]

np.vstack():
[[1 2 3]
 [4 5 6]]

np.dstack():
[[[ 7 10]]

 [[ 8 11]]

 [[ 9 12]]]
```

In [ ]:
```python
# Perform the searchsort method in Numpy array.
import numpy as np

# Create a NumPy array
arr = np.array([3, 1, 2, 5, 4])

# Sort the array
sorted_arr = np.sort(arr)

print("Original array:")
print(arr)
print("Sorted array:")
print(sorted_arr)
```

```
Original array:
[3 1 2 5 4]
Sorted array:
[1 2 3 4 5]
```

In [ ]:
```python
import numpy as np

# create a sample data for autonomous vehicle management
data = np.array([(1, 'Tesla', 'Model S', 2021, 50000),
                 (2, 'Toyota', 'Prius', 2022, 25000),
                 (3, 'BMW', 'i3', 2020, 30000),
                 (4, 'Nissan', 'Leaf', 2023, 35000),
                 (5, 'Ford', 'Mustang', 2022, 45000)],
                dtype=[('id', 'i4'), ('make', 'U10'), ('model', 'U10'), ('year', 'i

# display the structured array
print(data)
```

```
[(1, 'Tesla', 'Model S', 2021, 50000) (2, 'Toyota', 'Prius', 2022, 25000)
 (3, 'BMW', 'i3', 2020, 30000) (4, 'Nissan', 'Leaf', 2023, 35000)
 (5, 'Ford', 'Mustang', 2022, 45000)]
```

In [ ]:
```python
# Create Data frame using List and Dictionary.
import pandas as pd

# Create a list of data
data_list = [
    ['Alice', 25],
    ['Bob', 30],
    ['Charlie', 35],
    ['David', 40]
]

# Create a DataFrame from the list
df_list = pd.DataFrame(data_list, columns=['Name', 'Age'])

# Display the DataFrame
print(df_list)


# Create a dictionary of data
data_dict = {
    'Name': ['Alice', 'Bob', 'Charlie', 'David'],
    'Age': [25, 30, 35, 40]
}

# Create a DataFrame from the dictionary
df_dict = pd.DataFrame(data_dict)

# Display the DataFrame
print(df_dict)
```

```
      Name  Age
0    Alice   25
1      Bob   30
2  Charlie   35
3    David   40
      Name  Age
0    Alice   25
1      Bob   30
2  Charlie   35
3    David   40
```

In [ ]:
```python
# Create Data frame on your Domain area and perform the following operations to fin
# missing data from the dataset.
# • isnull()
# • notnull()
# • dropna()
# • fillna()
# • replace()
# • interpolate()
import pandas as pd

# create a dictionary of autonomous vehicle data
```

```
autonomous_vehicle_data = {'vehicle_id': [1, 2, 3, 4, 5],
                           'manufacturer': ['Tesla', 'Waymo', 'GM', 'Uber', 'Ford']
                           'model': ['Model S', 'Waymo One', 'Cruise AV', 'Volvo XC
                           'year': [2018, 2020, 2019, 2017, 2022],
                           'price': [50000, 100000, None, 70000, 60000]}

# create a data frame from the autonomous vehicle data dictionary
df = pd.DataFrame(autonomous_vehicle_data)

# display the data frame
print(df)

# check for missing values in the data frame
print(df.isnull())

# check for non-missing values in the data frame
print(df.notnull())

# drop rows with missing values
df.dropna(inplace=True)

# fill missing values with a specified value
df.fillna(0, inplace=True)

# replace specified values in the data frame
df.replace('Waymo', 'Alphabet', inplace=True)

# interpolate missing values in the data frame
df.interpolate(inplace=True)
```

```
   vehicle_id manufacturer       model  year     price
0           1        Tesla     Model S  2018   50000.0
1           2        Waymo   Waymo One  2020  100000.0
2           3           GM   Cruise AV  2019       NaN
3           4         Uber  Volvo XC90  2017   70000.0
4           5         Ford       F-150  2022   60000.0
   vehicle_id  manufacturer  model   year  price
0       False         False  False  False  False
1       False         False  False  False  False
2       False         False  False  False   True
3       False         False  False  False  False
4       False         False  False  False  False
   vehicle_id  manufacturer  model  year  price
0        True          True   True  True   True
1        True          True   True  True   True
2        True          True   True  True  False
3        True          True   True  True   True
4        True          True   True  True   True
C:\Users\sidha\AppData\Local\Temp\ipykernel_12644\2159026643.py:40: FutureWarning: D
ataFrame.interpolate with object dtype is deprecated and will raise in a future vers
ion. Call obj.infer_objects(copy=False) before interpolating instead.
  df.interpolate(inplace=True)
```

In [ ]:
```
import pandas as pd

# create a sample data frame
```

```
data = {
    'Make': ['Tesla', 'Toyota', 'BMW', 'Nissan', 'Ford'],
    'Model': ['Model 3', 'Prius', 'i3', 'Leaf', 'Mustang'],
    'Year': [2021, 2022, 2020, 2023, 2022],
    'Fuel Efficiency (km/L)': [None, 25, 20, None, 15]
}

df = pd.DataFrame(data)

# set the index to be hierarchical
df.set_index(['Make', 'Model'], inplace=True)

# display the data frame with hierarchical indexing
print(df)
```

```
               Year  Fuel Efficiency (km/L)
Make   Model
Tesla  Model 3  2021                     NaN
Toyota Prius    2022                    25.0
BMW    i3       2020                    20.0
Nissan Leaf     2023                     NaN
Ford   Mustang  2022                    15.0
```