1. Write a program to distinguish between Array Indexing and Fancy Indexing.

In [ ]:
```
import numpy as np

rows = int(input("Enter the number of rows: "))
cols = int(input("Enter the number of columns: "))

arr = np.empty((rows, cols), dtype=int)

for i in range(rows):
    for j in range(cols):
        arr[i, j] = int(input(f"{i+1} {j+1})"))

# Display the user's array
print("User's Array:")
print(arr)

#Array Indexing
element1 = arr[1, 1]

#Fancy Indexing
fancy_row= np.array([0,2])
fancy_col= np.array([0,1])
element2= arr[fancy_row,fancy_col]
print("Array Indexing:",element1)
print("Fancy Indexing:",element2 )
```

```
User's Array:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Array Indexing: 5
Fancy Indexing: [1 8]
```

2. Execute the 2D array Slicing.

In [ ]:
```
rows = int(input("Enter the number of rows: "))
columns = int(input("Enter the number of columns: "))
matrix = []
for i in range(rows):
    row = []
    for j in range(columns):
        element = int(input(f"{i+1}, {j+1}"))
        row.append(element)
    matrix.append(row)
print("The entered 2D array:")
for row in matrix:
    print(row)

start_row = int(input("Enter the starting row for slicing: "))
end_row = int(input("Enter the ending row (exclusive) for slicing: "))
start_column = int(input("Enter the starting column for slicing: "))
end_column = int(input("Enter the ending column (exclusive) for slicing: "))

if 0 <= start_row < end_row <= rows and 0 <= start_column < end_column <= columr
    sliced_portion = [row[start_column:end_column] for row in matrix[start_row:e
    print("The sliced portion:")
    for row in sliced_portion:
        print(row)
else:
    print("Invalid slicing indices.")
```

```
The entered 2D array:
[1, 2, 3]
[4, 5, 6]
[7, 8, 9]
The sliced portion:
[1, 2]
[4, 5]
```

3. Create the 5-Dimensional arrays using ‘ndmin’.

In [ ]:

```
import numpy as np

rows = int(input("Enter the number of rows: "))
cols = int(input("Enter the number of columns: "))
user_array = np.empty((rows, cols), dtype=int)

for i in range(rows):
    for j in range(cols):
        user_array[i, j] = int(input(f"{i+1}, {j+1}"))

print("Array:")
print(user_array)

print("5D Array:")
arr = np.array(user_array, ndmin=5)
print(arr)
```

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

4. Reshape the array from 1-D to 2-D array.

In [ ]:

```
import numpy as np

ele = int(input("Enter the number of elements in the 1-D array: "))

arr = np.empty(ele, dtype=int)

for i in range(ele):
    element = int(input(f"Enter element {i + 1}: "))
    arr[i] = element

num_rows = int(input("Enter the number of rows for the 2-D array: "))
num_cols = int(input("Enter the number of columns for the 2-D array: "))

arr2 = arr.reshape(num_rows, num_cols)
```

```
# Display the original 1-D array and the reshaped 2-D array
print("\nOriginal 1-D Array:")
print(arr)

print("\nReshaped 2-D Array:")
print(arr2)
```

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

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

5. Perform the Stack functions in Numpy arrays – Stack(), hstack(), vstack(), and dstack().

In [ ]:

```
import numpy as np

num_elements = int(input("Enter the number of elements for each array: "))
array1 = np.empty(num_elements, dtype=int)
array2 = np.empty(num_elements, dtype=int)
print("Enter elements for the first array:")
for i in range(num_elements):
    element = int(input(f"Enter element {i + 1} for the first array: "))
    array1[i] = element

print("\nEnter elements for the second array:")
for i in range(num_elements):
    element = int(input(f"Enter element {i + 1} for the second array: "))
    array2[i] = element

s = np.stack((array1, array2))
h = np.hstack((array1, array2))
v = np.vstack((array1, array2))
d = np.dstack((array1, array2))

print("\nStacked Arrays:")
print(s)

print("\nHorizontally Stacked Array:")
print(h)

print("\nVertically Stacked Array:")
print(v)

print("\nDepth-wise Stacked Array:")
print(d)
```

```
Enter elements for the first array:

Enter elements for the second array:

Stacked Arrays:
[[1 2 3]
 [1 2 3]]

Horizontally Stacked Array:
[1 2 3 1 2 3]

Vertically Stacked Array:
[[1 2 3]
 [1 2 3]]

Depth-wise Stacked Array:
[[[1 1]
  [2 2]
  [3 3]]]
```

6. Perform the searchsort method in Numpy array.

In [ ]:

```
import numpy as np

num_elements = int(input("Enter the number of elements in the array: "))

user_array = np.empty(num_elements, dtype=int)

for i in range(num_elements):
    element = int(input(f"Enter element {i + 1}: "))
    user_array[i] = element

sorted_array = np.sort(user_array)
search_value = int(input("Enter a value to search for: "))
index = np.searchsorted(sorted_array, search_value)
print("\nSorted Array:")
print(sorted_array)

print("\nIndex of Searched Value:", index)
```

```
Sorted Array:
[ 2  3  4  5  5  6  8  9 56]

Index of Searched Value: 1
```

7. Create Numpy Structured array using your domain features.

In [ ]:

```
import numpy as np

doc_dtype = np.dtype([
    ('document_id', np.int32),
    ('title', 'U100'),
    ('author', 'U50'),
    ('creation_date', 'M8[D]'),
    ('modification_date', 'M8[D]'),
    ('file_size', np.int64),
    ('keywords', 'U200'),
    ('category', 'U50'),
    ('is_archived', np.bool_),
])


document_data = np.array([], dtype=doc_dtype)

document = (1, "Sample Document", "Shubham Mishra", np.datetime64('2023-09-15'),
document_data = np.append(document_data, np.array([document], dtype=doc_dtype))


print(document_data)
```

```
[(1, 'Sample Document', 'Shubham Mishra', '2023-09-15', '2023-10-03', 1024, 'docu
ment, sample, demo', 'DemoFile', False)]
```

8. Create Data frame using List and Dictionary.

In [ ]:

```
import pandas as pd

data = [
    {"Book": "Harry Potter Series", "Author": "J.K. Rowling", "Publication Year"
    {"Book": "The Hobbit", "Author": "J.R.R. Tolkien", "Publication Year": 1937}
    {"Book": "The Lord of the Rings", "Author": "J.R.R. Tolkien", "Publication Y
    {"Book": "Python Crash Course", "Author": "Eric Matthes", "Publication Year"
    {"Book": "Clean Code", "Author": "Robert C. Martin", "Publication Year": 200
    {"Book": "Introduction to Machine Learning with Python", "Author": "Andreas 
]

df = pd.DataFrame(data)
print(df)
```

```
                                           Book  \
0                           Harry Potter Series   
1                                    The Hobbit   
2                         The Lord of the Rings   
3                           Python Crash Course   
4                                    Clean Code   
5  Introduction to Machine Learning with Python   

                               Author  Publication Year  
0                        J.K. Rowling              1997  
1                      J.R.R. Tolkien              1937  
2                      J.R.R. Tolkien              1954  
3                        Eric Matthes              2015  
4                    Robert C. Martin              2008  
5  Andreas C. Müller and Sarah Guido              2016  
```

9. Create Data frame on your Domain area and perform the following operations to find and eliminate the missing data from the dataset. • isnull() • notnull() • dropna() • fillna() • replace() • interpolate()

In [ ]:

```
import pandas as pd
import numpy as np

data = {
    'DocumentID': [1, 2, 3, 4, 5],
    'Title': ['Document A', 'Document B', 'Document C', None, 'Document E'],
    'Author': ['Author1', 'Author2', 'Author3', 'Author4', None],
    'Category': ['General', 'Technical', None, 'Demo', 'Educational'],
    'FileSize (KB)': [1024, None, 2048, None, 4096],
    'Keywords': ['keyword1', 'keyword2', None, 'keyword4', None],
}

df = pd.DataFrame(data)
print("Initial DataFrame:")
print(df)
null_data = df.isnull()
notnull_data = df.notnull()
dropped_data = df.dropna()
filled_data = df.fillna('Unknown')
replaced_data = filled_data.replace('Unknown', 'Not Specified')
interpolated_data = df.interpolate()

# Display the results of each operation
print("\nMissing Data (True indicates missing data):\n")
print(null_data)
print("\nNon-Missing Data:\n")
print(notnull_data)
print("\nDataFrame after Eliminating Rows with Missing Data:")
print(dropped_data)
print("\nDataFrame after Replacing Missing Values:")
print(filled_data)
print("\nDataFrame after Replacing Values:")
print(replaced_data)
print("\nDataFrame after Interpolating Missing Values:")
print(interpolated_data)
```

```
Initial DataFrame:
   DocumentID       Title   Author     Category  FileSize (KB)  Keywords
0           1  Document A  Author1      General         1024.0  keyword1
1           2  Document B  Author2    Technical            NaN  keyword2
2           3  Document C  Author3         None         2048.0      None
3           4        None  Author4         Demo            NaN  keyword4
4           5  Document E     None  Educational         4096.0      None

Missing Data (True indicates missing data):

   DocumentID  Title  Author  Category  FileSize (KB)  Keywords
0       False  False   False     False          False     False
1       False  False   False     False           True     False
2       False  False   False      True          False      True
3       False   True   False     False           True     False
4       False  False    True     False          False      True

Non-Missing Data:

   DocumentID  Title  Author  Category  FileSize (KB)  Keywords
0        True   True    True      True           True      True
1        True   True    True      True          False      True
2        True   True    True     False           True     False
3        True  False    True      True          False      True
4        True   True   False      True           True     False

DataFrame after Eliminating Rows with Missing Data:
   DocumentID       Title   Author Category  FileSize (KB)  Keywords
0           1  Document A  Author1  General         1024.0  keyword1

DataFrame after Replacing Missing Values:
   DocumentID       Title   Author     Category FileSize (KB)  Keywords
0           1  Document A  Author1      General        1024.0  keyword1
1           2  Document B  Author2    Technical       Unknown  keyword2
2           3  Document C  Author3      Unknown        2048.0   Unknown
3           4     Unknown  Author4         Demo       Unknown  keyword4
4           5  Document E  Unknown  Educational        4096.0   Unknown

DataFrame after Replacing Values:
   DocumentID          Title         Author       Category  FileSize (KB)  \
0           1     Document A        Author1        General         1024.0   
1           2     Document B        Author2      Technical  Not Specified   
2           3     Document C        Author3  Not Specified         2048.0   
3           4  Not Specified        Author4           Demo  Not Specified   
4           5     Document E  Not Specified    Educational         4096.0   

        Keywords  
0       keyword1  
1       keyword2  
2  Not Specified  
3       keyword4  
4  Not Specified  

DataFrame after Interpolating Missing Values:
   DocumentID       Title   Author     Category  FileSize (KB)  Keywords
0           1  Document A  Author1      General         1024.0  keyword1
1           2  Document B  Author2    Technical         1536.0  keyword2
2           3  Document C  Author3         None         2048.0      None
3           4        None  Author4         Demo         3072.0  keyword4
4           5  Document E     None  Educational         4096.0      None
```

```
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_21060\531832265.py:21: FutureWarnin
g: DataFrame.interpolate with object dtype is deprecated and will raise in a futu
re version. Call obj.infer_objects(copy=False) before interpolating instead.
  interpolated_data = df.interpolate()
```

10. Perform the Hierarchical Indexing in the above created dataset.

In [ ]:

```
df = pd.DataFrame(data)
df.set_index(['Author', 'Category'], inplace=True)
print("DataFrame with Hierarchical Index:")
print(df)
```

```
DataFrame with Hierarchical Index:
                     DocumentID       Title  FileSize (KB)  Keywords
Author  Category
Author1 General               1  Document A         1024.0  keyword1
Author2 Technical             2  Document B            NaN  keyword2
Author3 NaN                   3  Document C         2048.0      None
Author4 Demo                  4        None            NaN  keyword4
NaN     Educational           5  Document E         4096.0      None
```