Department of Engineering Sciences and Technology,

Second Year Btech in Computer Science

Project Based Learning-Python

Assignment - 21

Name - Paritosh kolwadkar

SRN – 31231313

Roll no – 39

Batch – D2

Problem statement :**Write a program to demonstrate various indexing techniques (e.g., accessing specific elements, rows, columns) and slicing operations to extract subarrays. Include examples of Boolean and fancy indexing.**

Prerequisite:

Knowledge of NumPy array creation and operations.

Understanding of indexing and slicing in NumPy.

Code:

```
import numpy as np



# Create a sample 2D NumPy array

array = np.array([[10, 20, 30, 40],

                  [50, 60, 70, 80],

                  [90, 100, 110, 120]])
```

```
print("Original Array:")
print(array)

# 1. Accessing Specific Elements
# Access element at row 1, column 2 (indexing starts from 0)
element = array[1, 2]
print("\nElement at row 1, column 2:", element)

# 2. Accessing Entire Rows and Columns
# Access all elements of row 0
row_0 = array[0, :]
print("\nRow 0:", row_0)

# Access all elements of column 2
col_2 = array[:, 2]
print("\nColumn 2:", col_2)

# 3. Slicing Operations (Extracting Subarrays)
# Extract a subarray of rows 0 to 1 and columns 1 to 2
subarray = array[0:2, 1:3]
print("\nSubarray (rows 0-1, columns 1-2):")
print(subarray)

# 4. Boolean Indexing (Filtering based on conditions)
# Create a boolean mask where we want values greater than 60
mask = array > 60
print("\nBoolean Mask (values > 60):")
print(mask)
```

```
# Use the mask to extract values greater than 60

filtered_values = array[mask]

print("\nFiltered Values (greater than 60):")

print(filtered_values)


# 5. Fancy Indexing (Selecting Specific Rows and Columns)

# Select rows 0 and 2, and columns 1 and 3

fancy_indexed = array[[0, 2], [1, 3]]

print("\nFancy Indexing (rows 0 and 2, columns 1 and 3):")

print(fancy_indexed)


# 6. Modifying Elements Using Indexing

# Change the element at row 1, column 1 to 999

array[1, 1] = 999

print("\nModified Array (element at row 1, column 1 changed to 999):")

print(array)
```

Explanation :

1. **Original Array:**
   - **The program begins by creating a 2D NumPy array called `array` with shape `(3, 4)` representing a 3x4 matrix.**
2. **Accessing Specific Elements:**
   - **`array[1, 2]`: Accesses the element at row 1, column 2 (values in NumPy arrays are indexed starting from 0).**
   - **The program prints the accessed element (70).**
3. **Accessing Entire Rows and Columns:**
   - **`array[0, :]`: This selects all the elements in row 0.**
   - **`array[:, 2]`: This selects all the elements in column 2.**
4. **Slicing Operations (Extracting Subarrays):**
   - **`array[0:2, 1:3]`: This slices the array to select rows 0 and 1, and columns 1 and 2. The result is a subarray of shape `(2, 2)`.**
5. **Boolean Indexing (Filtering Based on Conditions):**
   - **A boolean mask `array > 60` is created where each element is checked for the condition greater than 60. This results in a boolean array.**

- **The boolean mask is used to filter the elements from the original array that satisfy the condition.**

6. **Fancy Indexing (Selecting Specific Rows and Columns):**
   - **`array[[0, 2], [1, 3]]`: Fancy indexing is used to select rows 0 and 2, and columns 1 and 3. This is a more flexible indexing technique where you can select non-contiguous elements.**
7. **Modifying Elements Using Indexing:**
   - **`array[1, 1] = 999`: The program modifies the element at row 1, column 1, changing its value to 999.**

Output:

Original Array:

[[10 20 30 40]

 [50 60 70 80]

 [90 100 110 120]]

Element at row 1, column 2: 70

Row 0: [10 20 30 40]

Column 2: [30 70 110]

Subarray (rows 0-1, columns 1-2):

[[20 30]

 [60 70]]

Boolean Mask (values > 60):

[[False False False False]

 [False False True True]

 [True True True True]]

Filtered Values (greater than 60):

[70 80 90 100 110 120]

Fancy Indexing (rows 0 and 2, columns 1 and 3):

[20 40 100 120]

Modified Array (element at row 1, column 1 changed to 999):

[[10   20   30   40]

 [50 999   70   80]

 [90 100 110 120]]

Output Explained:

- **Accessing Specific Elements**: The program accesses and prints the element at position `(1, 2)` which is 70.
- **Accessing Rows and Columns**: It prints the entire row 0 and column 2.
- **Slicing Operations**: It extracts and displays the subarray from rows 0-1 and columns 1-2.
- **Boolean Indexing**: It prints a boolean mask and then filters out values greater than 60 from the original array.
- **Fancy Indexing**: It selects specific values from the array using row and column indices.
- **Modifying Elements**: It changes the value of the element at position `(1, 1)` to 999, and the modified array is displayed.

This program demonstrates how to perform various indexing and slicing operations in NumPy to manipulate and extract data from arrays.