PYTHON EXPERIMENT 10

Name : Gini Chacko
Roll : 8942
Class : SE Comps B

Aim: Write a Menu driven numpy program to:
1. Slicing multidimentional array
2. Flipping the order of the axes of multidimentional array.

CODE:

```
import numpy as np

while True:
    print("\n-----------------------------------------------------------")
    print("\n\n******Menu driven numpy program ******")
    print("1. Slicing multidimentional array")
    print("2. Flipping the order of the axes of multidimentional array.")
    print("3. Exit")
    choice = int(input("\nEnter your Choice : "))
    if choice==1:
        array2d = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])

        print("-" * 10)
        print(array2d[:, 0:2])  # 2nd and 3rd col

        print("-" * 10)
        print(array2d[1:3, 0:3])  # 2nd and 3rd row

        print("-" * 10)
        print(array2d[-1::-1, -1::-1])  # Reverse an array
```

```
    elif choice==2:
        array2d = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
        print(array2d)

        print("-" * 10)

        # Permute the dimensions of an array.
        arrayT = np.transpose(array2d)
        print(arrayT)

        print("-" * 10)

        # Flip array in the left/right direction.
        arrayFlr = np.fliplr(array2d)
        print(arrayFlr)

        print("-" * 10)

        # Flip array in the up/down direction.
        arrayFud = np.flipud(array2d)
        print(arrayFud)

        print("-" * 10)

        # Rotate an array by 90 degrees in the plane specified by axes.
        arrayRot90 = np.rot90(array2d)
        print(arrayRot90)

    elif choice==3:
        print("Exiting!!")
        print("\n----------------------------------------------------------")
        break

    else:
        print("Wrong Choice!")
```

OUTPUT:

```
PS C:\Users\Chacko> & python "c:/GINI/ENGG/2nd Year/Sem 4/Python/LABS/EXP 10/exp_10.py"

------------------------------------------------------------


******Menu driven numpy program ******
1. Slicing multidimentional array
2. Flipping the order of the axes of multidimentional array.
3. Exit

Enter your Choice : 1
----------
[[1 2]
 [4 5]
 [7 8]]
----------
[[4 5 6]
 [7 8 9]]
----------
[[9 8 7]
 [6 5 4]
 [3 2 1]]

------------------------------------------------------------
```

```
******Menu driven numpy program ******
1. Slicing multidimentional array
2. Flipping the order of the axes of multidimentional array.
3. Exit

Enter your Choice : 2
[[1 2 3]
 [4 5 6]
 [7 8 9]]
---------
[[1 4 7]
 [2 5 8]
 [3 6 9]]
---------
[[3 2 1]
 [6 5 4]
 [9 8 7]]
---------
[[7 8 9]
 [4 5 6]
 [1 2 3]]
---------
[[3 6 9]
 [2 5 8]
 [1 4 7]]

------------------------------------------------------------


******Menu driven numpy program ******
1. Slicing multidimentional array
2. Flipping the order of the axes of multidimentional array.
3. Exit

Enter your Choice : 5
Wrong Choice!

------------------------------------------------------------


******Menu driven numpy program ******
1. Slicing multidimentional array
2. Flipping the order of the axes of multidimentional array.
3. Exit

Enter your Choice : 3
Exiting!!

------------------------------------------------------------
```

POSTLAB QUESTIONS

1. Explain how do you change the shape of the Numpy array?

Ans:

The reshape() function is used to give a new shape to an array without changing its data.

Syntax:

```
numpy.reshape(a, newshape, order='C')
```

Parameter:

Name	Description	Required / Optional
a	Array to be reshaped.	Required
newshape	The new shape should be compatible with the original shape. If an integer, then the result will be a 1-D array of that length. One shape dimension can be -1. In this case, the value is inferred from the length of the array and remaining dimensions.	Required
order	Read the elements of a using this index order, and place the elements into the reshaped array using this index order. 'C' means to read / write the elements using C-like index order, with the last axis index changing fastest, back to the first axis index changing slowest. 'F' means to read / write the elements using Fortran-like index order, with the first index changing fastest, and the last index changing slowest. Note that the 'C' and 'F' options take no account of the memory layout of the underlying array, and only refer to the order of indexing. 'A' means to read / write the elements in Fortran-like index order if a is Fortran contiguous in memory, C-like order otherwise.	Optional

Return value:

reshaped_array : ndarray - This will be a new view object if possible; otherwise, it will be a copy. Note there is no guarantee of the memory layout (C- or Fortran-contiguous) of the returned array.

2. **How do you change the datatype of a Numpy array? Explain with a suitable example.**

Ans:

The Numpy array support a great variety of data types in addition to python's native data types. After an array is created, we can still modify the data type of the elements in the array, depending on our need. The two methods used for this purpose are **array.dtype** and **array.astype**

array.dtype

This method gives us the existing data type of the elements in the array. In the below example we declare an array and find its data types.

Example

```
import numpy as np

# Create a numpy array

a = np.array([21.23, 13.1, 52.1, 8, 255])

# Print the array

print(a)

# Print the array dat type

print(a.dtype)
```

array.astype

This method converts the existing array to a new array with desired data types.In the below example we take the given array and convert it to a variety of target data types.

Example

```
import numpy as np
# Create a numpy array
a = np.array([21.23, 13.1, 52.1, 8, 255])
# Print the array
print(a)
# Print the array dat type
print(a.dtype)
# Convert the array data type to int32
a_int = a.astype('int32')
print(a_int)
print(a_int.dtype)
# Convert the array data type to str
a_str = a.astype('str')
print(a_str)
print(a_str.dtype)
# Convert the array data type to complex
a_cmplx = a.astype('complex64')
print(a_cmplx)
print(a_cmplx.dtype)
```