```
# ********** NUMPY : Numerical Python **********
# Defines Multidimensional arrays
# It has functions for working in domains of Linear Algebra, Fourier Transformation & Matrices
# Fundamental Package for Scientific Computing in Python
# Arrays are in square braces [] lists but not coma seperated; e.g. [20 26 23 4 54 334]; lists can have multi
```

```
import numpy as np
```

```
# Creating Arrays & Matrices
arr = np.array([[20, 30, 40, 50], [10, 20, 30, 40], [30, 40, 40, 78]])
# print(arr)
# print(arr[0:2,0:2])
print(arr[2, 2:4])

```

```
[40 78]
```

```
np.shape(arr)                                                   # Tells the dimentions of the
np.size(arr)                                                    # number of rows multiply by
np.ndim(arr)                                                    # Dimension of the matrix
arr.dtype                                                       # Data Type of the elements o
len(arr)                                                        # Length of an matrix
arr.astype(float)                                               # Converts the data Type into
arr.astype(int)                                                 # Converts the data Type into
type(arr)
```

```
numpy.ndarray
```

```
# Mathemetical Operations & Functions on Arrays ; Either use operations or Functions
arr1 = np.array([[9, 20, 3], [44, 54, 46]])
arr2 = np.array([[6, 17, 81], [93, 10, 11]])
np.subtract(arr1, arr2)                                         # arr1 - arr2
np.add(arr1, arr2)                                              # arr1 + arr2
np.multiply(arr1, arr2)                                         # arr1 * arr2
np.divide(arr1, arr2)                                           # arr1 / arr2
np.power(arr1, 2)                                               # power
np.sqrt(arr1)
```

```
array([[3.        , 4.47213595, 1.73205081],
       [6.63324958, 7.34846923, 6.78232998]])
```

```
# Combining & Spliting Arrays
# Concatenate
np.concatenate([arr1, arr2],axis=0)                             # on x-axis
np.concatenate([arr1, arr2],axis=1)                             # on y-axis
np.hstack([arr1, arr2])                                         # Horizontal Stack
np.vstack([arr1, arr2])                                         # Vertical Stack
np.array_split(arr1, 2)                                         # split in 2 arrays
np.append(arr1,7)                                               # First converts into 1-dime
np.append(arr1,[7,60])                                          # First converts into 1-dime
```

```
array([ 9, 20,  3, 44, 54, 46,  7, 60])
```

```
# Adding & Removing Elements
a = np.array([[1, 2, 3, 4, 5, 6]])
np.append(a,7)                                                  # Append Items in Array
np.insert(a,1,7)                                                # Insert Items in an Array i
np.insert(arr1,1,[17,60], axis=1)                               # Created a new column on th
np.insert(arr1,1,[17,60,89], axis=0)                            # Created a new row on the g
np.insert(arr1,1,[17], axis=1)                                  # Created a new row on the g
np.insert(arr1,[1,2],[17,60,89], axis=0)                        # Inserting values on two di
np.delete(a,1)                                                  # Delete Items from an Array
np.delete(a,[1,2])                                              # Delete Items from an Array
np.delete(arr1,1,axis=0)                                        # Delete row on the given in
np.delete(arr1,1,axis=1)                                        # Delete column on the given
```

Show hidden output

```
# Search, Sort & Filter Arrays
ar = np.array([20, 13, 44, 15, 96, 7, 78, 29])
arr = np.array([[9, 20, 3], [44, 54, 46]])
ss = np.sort(ar)                        # Sorts an Array
np.sort(arr1)
np.where(ar == 90)
np.where(ar == 96)
np.where(ar%2 == 0)
np.searchsorted(ss,96)
```

```
np.searchsorted(ss,[96,15])

filtered_array = [True, False, True, False, True, False, True, False]
ar[filtered_array]

filtered_array = ar>20
ar[filtered_array]

filtered_array = ar%2 == 1
ar[filtered_array]
```

```
array([13, 15,  7, 29])
```

```
# Aggregating Functions in NUMPY
a = np.array([13,23, 20, 30, 27, 10])
np.sum(a)                                                    # Sum of all the value in th
np.min(a)                                                    # Minimum Value in the Array
np.max(a)                                                    # Maximum Value in the Array
np.mean(a)                                                   # Mean Value of the Array
np.median(a)                                                 # Median Value of the Array
np.average(a)                                                # Gives Average of the array
np.size(a)                                                   # Number of values/elements
np.cumsum(a)                                                 # Cumulative Sum
np.cumprod(a)                                                # Cumulative Product (multip
```

```
array([      13,      299,     5980,   179400,  4843800, 48438000])
```

```
# Aggregating Functions in NUMPY for 2 dimentional arrays
a = [110, 120, 200, 400, 600, 250]
b = [10, 12, 20, 40, 60, 25]
price = np.array([a])
quantity = np.array([b])
pricing = np.cumprod([price, quantity], axis = 0)
np.sum(pricing[1])
```

```
np.int64(64790)
```

```
import statistics as stats # for Mode
```

```
# Statistical Functions in Arrays. -------- # Example
import numpy as np
baked_food = [200, 300, 150, 100, 130, 280, 280]
a = np.array([baked_food])

np.mean(a)                                                   # Mean : sum of all values /
np.median(a)                                                 # Median : First sorts it an
np.average(a)                                                # Gives Average of the array
np.size(a)
np.var(a)                                                    # Variance : first find the
np.std(a)                                                    # Standard Deviation ; take
stats.mode(baked_food)                                       # Mode: Most occuring value

tobacco_consumption = [30, 50, 10, 30, 50, 40]
deaths = [100, 120, 70, 100, 120, 112]

np.corrcoef([tobacco_consumption, deaths])                   # coefficient of correlation
```

Show hidden output

```
# ********************************************** Pandas **********************************************
# Pandas is Python Package providing fast, flexible & Expressive data structures designed to make working wi
# It has functions for analyzing, cleaning, exploring, & manipulating data.
# The name pandas refered to as "Panel Data"
# It provide easy handeling of missing data
# Columns can be inserted or deleted from the data frames & higher dimensional objects.
# Autometically alligns the data for you in computations
# Powerful and functional group of functionality
# It provide Intellegent label-based slicing, fancy indexing, and sub-setting of large data sets
# Pandas provide flexible reshaping and pivoting of data sets.

# The best way to think about the pandas data structures is a flexible containers for lower dimensional data
# Pandas series is one dimensional labeled array capable of holding data of any datatype (integer, float, st
# Panda series is nothing but a column in excel.
# the object supports both integer and label based indexing & provide a host of methods for performing opera
# Panda DataFrame is two-dimensional size-muteable, potentionaly heterogeneous tabular data structure with l
# Pandas DataFrame consists of three principle components the data, rows, & Columns.

```

```
import pandas as pd
```

```
# ------------------------------ creation of Data Frames in pandas ------------------------------
data = {"name": ["john", "saba", "mirza", "alex"],
        "age": [25, 27, 28, 20],
        "salary": [300000, 400000, 350000, 200000]}
df = pd.DataFrame(data)                                                    # Create Data Frame (in tabu

# Other methods of reading data from the different files
# df = pd.read_csv("name of the file.csv")
# df = pd.read_csv("link of the file from the folder / name of the file.csv")   # change \ to / (if you have
# df = pd.read_excel("file name.xlsx")

df
```

Show hidden output

```
# ------------------------------ Example ------------------------------
s = pd.Series([1, 2, None, 4, float('nan'), 6])
df = pd.read_csv('FoP.csv')
df["Price"] = pd.to_numeric(df["Price"], errors="coerce")                  # invalid values → become NaN
df["Qty"] = df["Qty"].astype("int64")                                      # Convert integer columns

# Remove rows that have NaN after conversion
df = df.dropna(subset=["Price", "Date"])
# Only remove NaN from some columns, not whole dataframe

# Check unique formats in a column
df["Date"].head()
df["Date"].sample(5)
df["Date"].str.len().unique()

# Removing Column
df = df.drop(columns=["ColumnName"])
# Removing Multiple Columns
df = df.drop(columns=["Age","Address","Phone"])
# Remove by Index
df = df.drop(df.columns[3], axis=1)                                        # remove 4th column
df = df.drop(df.columns[[1,3]], axis=1)                                    # remove 2nd and 4th column

```

```
# ------------------------------ Exploring data in pandas ------------------------------
df = pd.read_excel("Practice Session - DA.xlsx")
print("first five values of the data", df.head())
df.tail()                                                                  # Show last 5 rows
df.head(10)                                                                # Show first 10 rows
df.tail(10)                                                                # Show last 10 rows
df.info()                                                                  # Tells overall information
df.describe()
df.isnull()
df.isnull().sum()
```

Show hidden output

```
# ------------------------------ Handling Duplicate Values in Pandas ------------------------------
df.duplicated()
df["Hospital ID"].duplicated()
df["Hospital ID"].duplicated().sum()
df.drop_duplicates("Hospital ID")
```

Show hidden output

```
# ------------------------------ Working with Missing Data in Pandas ------------------------------
df.isnull()
df.isnull().sum()
#df.dropna()
# df.replace(np.nan,"hi")
df.fillna(0)
df["Avg. Treatment Cost (PKR)"].replace(np.nan,df["Avg. Treatment Cost (PKR)"].mean())
df["Avg. Treatment Cost (PKR)"].fillna(df["Avg. Treatment Cost (PKR)"].mean())
# df["Avg. Treatment Cost (PKR)"].mean()
df.fillna(method="ffill")                                                  # Forward Fill
df.fillna(method="bfill")                                                  # Backward Fill
```

Show hidden output

```
# ------------------------------ Columns Transformation in Pandas ------------------------------
df = pd.read_excel("ESD.xlsx")                                             # Employee Sample Data
df.head()
df.loc[(df["Bonus %"] == 0), "Gets Bonus"] = "No Bonus"                    # Creating new column using
```

```
df.loc[(df["Bonus %"] > 0), "Gets Bonus"] = "Bonus"
df.head()
```

Show hidden output

```
df = pd.read_excel("practice.xlsx")
df["Full Name"] = df["First Name"] + df["Last Name"]
df["Full Name"] = df["First Name"] + " " + df["Last Name"]
df["Full Name"] = df["First Name"] + " " + df["Last Name"].str.upper()
df["Full Name"] = df["First Name"] + " " + df["Last Name"].str.lower()
df["Full Name"] = df["First Name"] + " " + df["Last Name"].str.title()
df["Full Name"] = df["First Name"] + " " + df["Last Name"].str.capitalize()
df["Full Name"] = df["First Name"].str.capitalize() + " " + df["Last Name"].str.capitalize()
```

Show hidden output

```
# ------------------------------ Summarizing Data (Group by) in Pandas ------------------------------
df = pd.read_excel("ESD.xlsx")
df
df.groupby("Department").agg({"EEID":"count"})
df.groupby(["Department", "Gender"]).agg({"EEID":"count"})

df.groupby("Country").agg({"Age":"mean"})
df.groupby(["Country", "Gender"]).agg({"Annual Salary":"max"})
df.groupby(["Country", "Gender"]).agg({"Annual Salary":"min"})
df.groupby(["Country", "Gender"]).agg({"Annual Salary":"max" , "Age" : "min"})
df.groupby("Country").agg({"Age":"mean", "Annual Salary":"mean"})
```

```
# ------------------------------ Merge, Join, & Concatenate in Pandas ------------------------------
import pandas as pd
data1 = {"EID": ["E01", "E02", "E03", "E04"],
        "Names": ["john", "saba", "alex", "mirza"]}

data2 = {"EID": ["E01", "E05", "E03", "E04"],
          "Salary": [300000, 400000, 350000, 200000]}
df1 = pd.DataFrame(data1)
df2 = pd.DataFrame(data2)
print("df1:",df1)
print()
print("df2:",df2)
df3 = pd.merge(df1, df2, on = "EID")
# df3 = pd.merge(left = df1, right = df2, on = "EID" , how = "left")
# df3 = pd.merge(left = df1, right = df2, on = "EID" , how = "right")
# df3 = pd.merge(left = df1, right = df2, on = "EID" , how = "outer")
df3 = pd.merge(left = df1, right = df2, on = "EID" , how = "inner")
df3
```

Show hidden output

```
import pandas as pd
data1 = {"EID": ["E01", "E02", "E03", "E04"],
        "Names": ["john", "saba", "alex", "mirza"],
         "Salary": [10000, 60000, 90000, 40000]}

data2 = {"EID": ["E05", "E06", "E07", "E08"],
         "Names": ["aser", "ahcene", "inder", "laurent"],
          "Salary": [300000, 400000, 350000, 200000]}
df1 = pd.DataFrame(data1)
df2 = pd.DataFrame(data2)
df3 = pd.concat([df1, df2], axis = 0)
df3
```

Show hidden output

```
# ------------------------------ Compare Data Frames in Pandas ------------------------------
```