Sol 1)

```
Import numpy as np
arr1d = np.arange(1, 11)
arr2d = np.arange(1, 10).reshape(3, 3)
arr3d = np.random.rand(3, 5, 3)
For arr in [arr1d, arr2d, arr3d]:
   Print(“Shape:”, arr.shape)
   Print(“Size:”, arr.size)
   Print(“Data type:”, arr.dtype)
   Print(“-“ * 30)
```

Sol2)

```
Data = np.array([10, 20, 30, 40, 50, 60, 70, 80, 90])

Print(“First 3 elements:”, data[:3])
Print(“Alternate elements:”, data[::2])
Print(“Reversed array:”, data[::-1])
```

Sol3)

```
A = np.random.randint(1, 21, 5)
B = np.random.randint(1, 21, 5)
Print(“A:”, A)
Print(“B:”, B)
Print(“Add:”, A + B)
Print(“Subtract:”, A – B)
Print(“Multiply:”, A * B)
Print(“Divide:”, A / B)
```

```
Print("Dot Product:", np.dot(A, B))
Print("Mean:", np.mean(A))
Print("Median:", np.median(A))
Print("Std Dev:", np.std(A))
Print("Variance:", np.var(A))
Print("Max in B:", np.max(B), "at index", np.argmax(B))
Print("Min in B:", np.min(B), "at index", np.argmin(B))
```

Sol 4)

```
Array = np.arange(1, 13)

Reshaped_2d = array.reshape(4, 3)
Reshaped_3d = array.reshape(2, 2, 3)
Transposed = reshaped_2d.T

Print("Reshaped 2D:\n", reshaped_2d)
Print("Reshaped 3D:\n", reshaped_3d)
Print("Transposed 2D:\n", transposed)
Print("Transposed Shape:", transposed.shape)
```

Sol5)

```
Arr = np.random.randint(10, 51, 15)
Print("Original:", arr)
Print("Elements > 25:", arr[arr > 25])
Arr[arr < 30] = 0
Print("Replaced <30 with 0:", arr)
```

```
Count_div_5 = np.sum(arr % 5 == 0)
Print(“Count divisible by 5:”, count_div_5)
```

Sol6)

```
Import numpy as np

# Identity matrix of size 4x4
Identity_matrix = np.eye(4)
Print(“Identity Matrix:\n”, identity_matrix)
Random_array = np.random.randint(1, 101, 20)
Sorted_array = np.sort(random_array)
Largest_five = sorted_array[-5:]
Print(“Random Array:”, random_array)
Print(“Sorted Array:”, sorted_array)
Print(“Five Largest Elements:”, largest_five)
```

Sol7)

```
Import time
A = np.random.rand(100, 100)
B = np.random.rand(100, 100)
Start_time = time.time();
Product = np.dot(A, B)
Try:
   Det = np.linalg.det(product)
   Inv = np.linalg.inv(product)
   Print(“Determinant:”, det)
```

```
   Print("Inverse (first 5x5 block):\n", inv[:5, :5])
Except np.linalg.LinAlgError:
   Print("Matrix is singular and not invertible.")

End_time = time.time()
Elapsed_time = end_time – start_time

Print("Time taken for operations: {:.4f} seconds".format(elapsed_time))
```

Part 2

Sol 1)

```
Import pandas as pd
Data = [25, 30, 35, 40, 45]
Labels = ['A', 'B', 'C', 'D', 'E']
Series = pd.Series(data, index=Labels);
Print("First three elements:\n", series[:3])
Print("Mean:", series.mean())
Print("Median:", series.median())
Print("Standard Deviation:", series.std())
```

Sol2)

```
Info = {
   'Name': ['Alice', 'Bob', 'Carol', 'David', 'Eve'],
   'Age': [20, 22, 19, 21, 20],
   'Gender': ['Female', 'Male', 'Female', 'Male', 'Female'],
```

```
    'Marks': [85, 78, 92, 74, 88]
}
Df = pd.DataFrame(info)
Print("First two rows:\n", df.head(2))
Print("Column Names:", df.columns.tolist())
Print("Data Types:\n", df.dtypes)
Print("Summary Statistics:\n", df.describe())
Df['Passed'] = df['Marks'] >= 80
Print("Updated DataFrame with 'Passed':\n", df)
```

Sol3)

```
Print("Selected columns:\n", df[['Name', 'Marks']])
Print("Students scoring above 80:\n", df[df['Marks'] > 80])
Topper = df[df['Marks'] == df['Marks'].max()]
Print("Top scorer:\n", topper)
```

Sol4)

```
Import pandas as pd
Import numpy as np
Data = {
    'Name': ['Ankit', 'Priya', 'Rahul', 'Neha', 'Amit'],
    'Age': [20, 21, 19, 22, 20],
    'Gender': ['Male', 'Female', 'Male', 'Female', 'Male'],
    'Marks': [85, 90, 88, 92, 87]
}
```

```
Df = pd.DataFrame(data)
Df.loc[1, ‘Marks’] = None
Df.loc[4, ‘Age’] = None
Missing_info = df.isnull()
Df[‘Marks’].fillna(df[‘Marks’].mean(), inplace=True)
Df.dropna(subset=[‘Age’], inplace=True)
Print(“Modified DataFrame:”)
Print(df)
```

Sol5)

```
Grouped = df.groupby(‘Gender’)[[‘Age’, ‘Marks’]].mean()
Gender_counts = df[‘Gender’].value_counts()

Print(“\nMean Age and Marks by Gender:”)
Print(grouped)

Print(“\nCount of Students by Gender:”)
Print(gender_counts)
```

Sol6)

```
Df.to_csv(“students_data.csv”, index=False)
Df_new = pd.read_csv(“students_data.csv”)
Print(“\nFirst 5 rows of the new DataFrame:”)
Print(df_new.head())
```

Sol6)

```
Df.to_csv(“students_data.csv”, index=False)
Df_new = pd.read_csv(“students_data.csv”)
Print(“\nFirst 5 rows of the new DataFrame:”)
Print(df_new.head())
```

Sol7)

```
Import seaborn as sns
Import matplotlib.pyplot as plt
Df_eda = sns.load_dataset(‘titanic’)
Print(“\nSummary statistics:”)
Print(df_eda.describe())
Print(“\nMissing values:”)
Print(df_eda.isnull().sum())
Sns.countplot(x=’sex’, hue=’survived’, data=df_eda)
Plt.title(“Survival count by Gender”)
Plt.show()
Sns.histplot(df_eda[‘age’].dropna(), kde=True)
Plt.title(“Age Distribution”)
Plt.show()
```

Analysis:

- Age and Fare have wide distributions.
- Many missing values in Age and Cabin columns.
- Female passengers had higher survival rates.