```
import pandas as pd
import numpy as np
import seaborn as sns
```

1.) Loading the dataset

```
df=pd.read_csv('/content/sample_data/kidneyDisease.csv')
```

2.) Inspect/know the data (EDA)

```
df.shape
```

```
(4116, 19)
```

```
# df.shape[0]
# df.shape[1]
```

```
df.dtypes
```

	0
SUBDIVISION	object
YEAR	int64
JAN	float64
FEB	float64
MAR	float64
APR	float64
MAY	float64
JUN	float64
JUL	float64
AUG	float64
SEP	float64
OCT	float64
NOV	float64
DEC	float64
ANNUAL	float64
Jan-Feb	float64
Mar-May	float64
Jun-Sep	float64
Oct-Dec	float64

dtype: object

```
df.head()
```

	SUBDIVISION	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	I
0	ANDAMAN & NICOBAR ISLANDS	1901	49.2	87.1	29.2	2.3	528.8	517.5	365.1	481.1	332.6	388.5	55
1	ANDAMAN & NICOBAR ISLANDS	1902	0.0	159.8	12.2	0.0	446.1	537.1	228.9	753.7	666.2	197.2	35
2	ANDAMAN & NICOBAR ISLANDS	1903	12.7	144.0	0.0	1.0	235.1	479.9	728.4	326.7	339.0	181.2	28
3	ANDAMAN & NICOBAR ISLANDS	1904	9.4	14.7	0.0	202.4	304.5	495.1	502.0	160.1	820.4	222.2	30
4	ANDAMAN & NICOBAR ISLANDS	1905	1.3	0.0	3.3	26.9	279.5	628.7	368.7	330.5	297.0	260.7	2

```
#df.head(1)
```

```
df.tail()
```

	SUBDIVISION	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT
4111	LAKSHADWEEP	2011	5.1	2.8	3.1	85.9	107.2	153.6	350.2	254.0	255.2	117.4
4112	LAKSHADWEEP	2012	19.2	0.1	1.6	76.8	21.2	327.0	231.5	381.2	179.8	145.9
4113	LAKSHADWEEP	2013	26.2	34.4	37.5	5.3	88.3	426.2	296.4	154.4	180.0	72.8
4114	LAKSHADWEEP	2014	53.2	16.1	4.4	14.9	57.4	244.1	116.1	466.1	132.2	169.2
4115	LAKSHADWEEP	2015	2.2	0.5	3.7	87.1	133.1	296.6	257.5	146.4	160.4	165.4

```
# df.tail(1)
df.describe()
```

	YEAR	JAN	FEB	MAR	APR	MAY	
count	4116.000000	4112.000000	4113.000000	4110.000000	4112.000000	4113.000000	4111.
mean	1958.218659	18.957320	21.805325	27.359197	43.127432	85.745417	230.
std	33.140898	33.585371	35.909488	46.959424	67.831168	123.234904	234.
min	1901.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.
25%	1930.000000	0.600000	0.600000	1.000000	3.000000	8.600000	70.
50%	1958.000000	6.000000	6.700000	7.800000	15.700000	36.600000	138.
75%	1987.000000	22.200000	26.800000	31.300000	49.950000	97.200000	305.
max	2015.000000	583.700000	403.500000	605.600000	595.100000	1168.600000	1609.

```
df.describe(include='object')
```

	SUBDIVISION
count	4116
unique	36
top	WEST MADHYA PRADESH
freq	115

```
df.describe(include='all')
```

	SUBDIVISION	YEAR	JAN	FEB	MAR	APR	
count	4116	4116.000000	4112.000000	4113.000000	4110.000000	4112.000000	411
unique	36	NaN	NaN	NaN	NaN	NaN	
top	WEST MADHYA PRADESH	NaN	NaN	NaN	NaN	NaN	
freq	115	NaN	NaN	NaN	NaN	NaN	
mean	NaN	1958.218659	18.957320	21.805325	27.359197	43.127432	8
std	NaN	33.140898	33.585371	35.909488	46.959424	67.831168	12
min	NaN	1901.000000	0.000000	0.000000	0.000000	0.000000	
25%	NaN	1930.000000	0.600000	0.600000	1.000000	3.000000	
50%	NaN	1958.000000	6.000000	6.700000	7.800000	15.700000	3
75%	NaN	1987.000000	22.200000	26.800000	31.300000	49.950000	9
max	NaN	2015.000000	583.700000	403.500000	605.600000	595.100000	116

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4116 entries, 0 to 4115
Data columns (total 19 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   SUBDIVISION  4116 non-null   object 
 1   YEAR         4116 non-null   int64  
 2   JAN          4112 non-null   float64
 3   FEB          4113 non-null   float64
 4   MAR          4110 non-null   float64
 5   APR          4112 non-null   float64
 6   MAY          4113 non-null   float64
 7   JUN          4111 non-null   float64
 8   JUL          4109 non-null   float64
 9   AUG          4112 non-null   float64
 10  SEP          4110 non-null   float64
 11  OCT          4109 non-null   float64
 12  NOV          4105 non-null   float64
 13  DEC          4106 non-null   float64
 14  ANNUAL       4090 non-null   float64
 15  Jan-Feb      4110 non-null   float64
 16  Mar-May      4107 non-null   float64
 17  Jun-Sep      4106 non-null   float64
 18  Oct-Dec      4103 non-null   float64
dtypes: float64(17), int64(1), object(1)
memory usage: 611.1+ KB
```

3) Handling missing values

```
#find/check missing values
print(df.isnull().sum())
```

```
SUBDIVISION     0
YEAR            0
JAN             4
FEB             3
MAR             6
APR             4
MAY             3
JUN             5
JUL             7
AUG             4
SEP             6
OCT             7
NOV            11
DEC            10
ANNUAL         26
Jan-Feb         6
Mar-May         9
Jun-Sep        10
Oct-Dec        13
dtype: int64
```

```
# missingValueCount=df.isnull().sum()
# missingValueCount[0:10]
df_droppedRows=df.dropna()
df_droppedRows.shape[0]
```

```
158
```

```
df_droppedCols=df.dropna(axis=1)
df_droppedCols.shape[1]
```

```
2
```

```
df_dropped50percentage=df.dropna(axis=1, thresh=int(0.5*len(df)),inplace=True)
df_dropped50percentage
df_filled=df.fillna(0)
print(df_filled.isnull().sum())
```

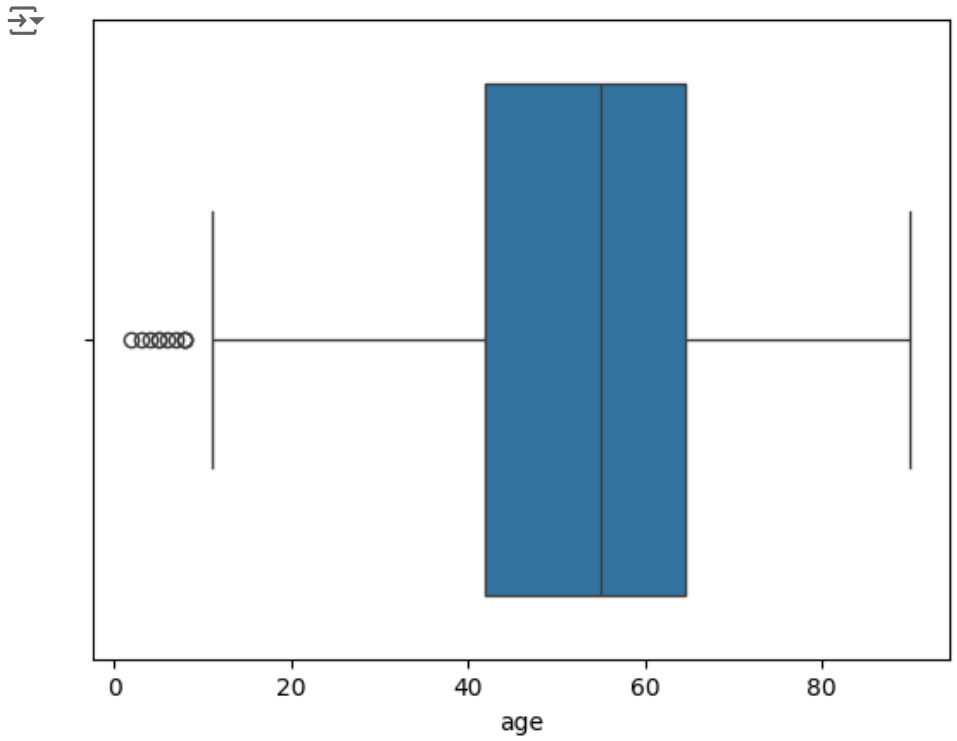
```
SUBDIVISION    0
YEAR           0
JAN            0
FEB            0
MAR            0
APR            0
MAY            0
JUN            0
JUL            0
AUG            0
SEP            0
OCT            0
NOV            0
DEC            0
ANNUAL         0
Jan-Feb        0
Mar-May        0
Jun-Sep        0
Oct-Dec        0
dtype: int64
```

```
for cols in df.columns:
    if df[cols].dtype=='object':
      df[cols].fillna(df[cols].mode()[0])
    else:
      df[cols].fillna(df[cols].mean())
print(df.isnull().sum())
```

```
SUBDIVISION     0
YEAR            0
JAN             4
FEB             3
MAR             6
APR             4
MAY             3
JUN             5
JUL             7
AUG             4
SEP             6
OCT             7
NOV            11
DEC            10
ANNUAL         26
Jan-Feb         6
Mar-May         9
Jun-Sep        10
Oct-Dec        13
dtype: int64
```

4) Handling outliers

```
import seaborn as sns
import matplotlib.pyplot as plt
sns.boxplot(x=df['age'])
plt.show()
```




```
# Calculate the interquartile Range (IQR)
Q1=df['age'].quantile(0.25)
Q3=df['age'].quantile(0.75)
IQR=Q3-Q1
df=df[~((df['age'] < (Q1 - 1.5*IQR)) | (df['age']> (Q3+1.5 *IQR)))]
```

5) Handling inconsistent values

```
df['rbc']=df['rbc'].str.lower()
df['rbc'].replace({'n': 'normal', 'ab':'abnormal'})
df
```

	id	age	bp	sg	al	su	rbc	pc	pcc	ba	...	pcv	wc	rc	htn	dı
0	0	48.0	80.0	1.020	1.0	0.0	NaN	normal	notpresent	notpresent	...	44	7800	5.2	yes	ye
2	2	62.0	80.0	1.010	2.0	3.0	normal	normal	notpresent	notpresent	...	31	7500	NaN	no	ye
3	3	48.0	70.0	1.005	4.0	0.0	normal	abnormal	present	notpresent		32	6700	3.9	yes	no

6) Validation checks