DSBDAL2_DataWranglingII

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:
```
df = pd.read_csv("A1.csv")
df
```

Out[2]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	1	Student_1	100	62	73.0	92.0	96
1	2	Student_2	72	97	82.0	NaN	78
2	3	Student_3	100	88	71.0	99.0	-94
3	4	Student_4	72	99	NaN	84.0	86
4	5	Student_5	97	70	84.0	70.0	86
...	...	...	...	...	...	...	...
996	997	Student_997	88	68	84.0	66.0	98
997	998	Student_998	61	96	62.0	84.0	83
998	999	Student_999	72	76	90.0	72.0	90
999	1000	Student_1000	68	87	100.0	76.0	79
1000	1	Student_1	100	62	73.0	92.0	96

1001 rows × 7 columns

In [3]:
```
df.head()
```

Out[3]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	1	Student_1	100	62	73.0	92.0	96
1	2	Student_2	72	97	82.0	NaN	78
2	3	Student_3	100	88	71.0	99.0	-94
3	4	Student_4	72	99	NaN	84.0	86
4	5	Student_5	97	70	84.0	70.0	86

In [4]:
```
df.shape
```

Out[4]:
```
(1001, 7)
```

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1001 entries, 0 to 1000
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Roll No     1001 non-null   int64
 1   Name        1001 non-null   object
 2   Subject 1   1001 non-null   int64
```

```
 3   Subject 2   1001 non-null   int64
 4   Subject 3   1000 non-null   float64
 5   Subject 4   1000 non-null   float64
 6   Attendance  1001 non-null   int64
dtypes: float64(2), int64(4), object(1)
memory usage: 54.9+ KB
```

Handling Missing Data

isnull()

In [6]:
```
df.isnull()
```

Out[6]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	False	False	False	False	False	False	False
1	False	False	False	False	False	True	False
2	False	False	False	False	False	False	False
3	False	False	False	False	True	False	False
4	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...
996	False	False	False	False	False	False	False
997	False	False	False	False	False	False	False
998	False	False	False	False	False	False	False
999	False	False	False	False	False	False	False
1000	False	False	False	False	False	False	False

1001 rows × 7 columns

In [7]:
```
## to find total null values is each column
df.isnull().sum()
```

Out[7]:
```
Roll No       0
Name          0
Subject 1     0
Subject 2     0
Subject 3     1
Subject 4     1
Attendance    0
dtype: int64
```

notnull()

In [8]:
```
df.notnull()
```

Out[8]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	True	True	True	True	True	True	True
1	True	True	True	True	True	False	True
2	True	True	True	True	True	True	True
3	True	True	True	True	False	True	True
4	True	True	True	True	True	True	True

...	...	...	...	...	...	...	...
996	True	True	True	True	True	True	True
997	True	True	True	True	True	True	True
998	True	True	True	True	True	True	True
999	True	True	True	True	True	True	True
1000	True	True	True	True	True	True	True

1001 rows × 7 columns

In [9]:
```
df.notnull().sum()
```

Out[9]:
```
Roll No       1001
Name          1001
Subject 1     1001
Subject 2     1001
Subject 3     1000
Subject 4     1000
Attendance    1001
dtype: int64
```

Handling the missing values

fillna()

In [10]:
```
df.fillna(0,inplace=True)
df
```

Out[10]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	1	Student_1	100	62	73.0	92.0	96
1	2	Student_2	72	97	82.0	0.0	78
2	3	Student_3	100	88	71.0	99.0	-94
3	4	Student_4	72	99	0.0	84.0	86
4	5	Student_5	97	70	84.0	70.0	86
...	...	...	...	...	...	...	...
996	997	Student_997	88	68	84.0	66.0	98
997	998	Student_998	61	96	62.0	84.0	83
998	999	Student_999	72	76	90.0	72.0	90
999	1000	Student_1000	68	87	100.0	76.0	79
1000	1	Student_1	100	62	73.0	92.0	96

1001 rows × 7 columns

In [11]:
```
df = pd.read_csv("A1.csv")
df
```

Out[11]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	1	Student_1	100	62	73.0	92.0	96
1	2	Student_2	72	97	82.0	NaN	78

2	3	Student_3	100	88	71.0	99.0	-94
3	4	Student_4	72	99	NaN	84.0	86
4	5	Student_5	97	70	84.0	70.0	86
...	...	...	...	...	...	...	...
996	997	Student_997	88	68	84.0	66.0	98
997	998	Student_998	61	96	62.0	84.0	83
998	999	Student_999	72	76	90.0	72.0	90
999	1000	Student_1000	68	87	100.0	76.0	79
1000	1	Student_1	100	62	73.0	92.0	96

1001 rows × 7 columns

In [12]:
```
df.fillna(50,inplace=True)
df
```

Out[12]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	1	Student_1	100	62	73.0	92.0	96
1	2	Student_2	72	97	82.0	50.0	78
2	3	Student_3	100	88	71.0	99.0	-94
3	4	Student_4	72	99	50.0	84.0	86
4	5	Student_5	97	70	84.0	70.0	86
...	...	...	...	...	...	...	...
996	997	Student_997	88	68	84.0	66.0	98
997	998	Student_998	61	96	62.0	84.0	83
998	999	Student_999	72	76	90.0	72.0	90
999	1000	Student_1000	68	87	100.0	76.0	79
1000	1	Student_1	100	62	73.0	92.0	96

1001 rows × 7 columns

In [13]:
```
df = pd.read_csv("A1.csv")
df
```

Out[13]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	1	Student_1	100	62	73.0	92.0	96
1	2	Student_2	72	97	82.0	NaN	78
2	3	Student_3	100	88	71.0	99.0	-94
3	4	Student_4	72	99	NaN	84.0	86
4	5	Student_5	97	70	84.0	70.0	86
...	...	...	...	...	...	...	...
996	997	Student_997	88	68	84.0	66.0	98
997	998	Student_998	61	96	62.0	84.0	83
998	999	Student_999	72	76	90.0	72.0	90
999	1000	Student_1000	68	87	100.0	76.0	79

1000	1	Student_1	100	62	73.0	92.0	96

1001 rows × 7 columns

In [14]:
```
df.fillna(method='pad')
```

Out[14]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	1	Student_1	100	62	73.0	92.0	96
1	2	Student_2	72	97	82.0	92.0	78
2	3	Student_3	100	88	71.0	99.0	-94
3	4	Student_4	72	99	71.0	84.0	86
4	5	Student_5	97	70	84.0	70.0	86
...	...	...	...	...	...	...	...
996	997	Student_997	88	68	84.0	66.0	98
997	998	Student_998	61	96	62.0	84.0	83
998	999	Student_999	72	76	90.0	72.0	90
999	1000	Student_1000	68	87	100.0	76.0	79
1000	1	Student_1	100	62	73.0	92.0	96

1001 rows × 7 columns

In [15]:
```
df = pd.read_csv("A1.csv")
df
```

Out[15]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	1	Student_1	100	62	73.0	92.0	96
1	2	Student_2	72	97	82.0	NaN	78
2	3	Student_3	100	88	71.0	99.0	-94
3	4	Student_4	72	99	NaN	84.0	86
4	5	Student_5	97	70	84.0	70.0	86
...	...	...	...	...	...	...	...
996	997	Student_997	88	68	84.0	66.0	98
997	998	Student_998	61	96	62.0	84.0	83
998	999	Student_999	72	76	90.0	72.0	90
999	1000	Student_1000	68	87	100.0	76.0	79
1000	1	Student_1	100	62	73.0	92.0	96

1001 rows × 7 columns

In [16]:
```
df.fillna(method='bfill')
```

Out[16]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	1	Student_1	100	62	73.0	92.0	96
1	2	Student_2	72	97	82.0	99.0	78
2	3	Student_3	100	88	71.0	99.0	-94

3	4	Student_4	72	99	84.0	84.0	86
4	5	Student_5	97	70	84.0	70.0	86
...	...	...	...	...	...	...	...
996	997	Student_997	88	68	84.0	66.0	98
997	998	Student_998	61	96	62.0	84.0	83
998	999	Student_999	72	76	90.0	72.0	90
999	1000	Student_1000	68	87	100.0	76.0	79
1000	1	Student_1	100	62	73.0	92.0	96

1001 rows × 7 columns

interpolate()

In [17]:
```
df = pd.read_csv("A1.csv")
df
```

Out[17]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	1	Student_1	100	62	73.0	92.0	96
1	2	Student_2	72	97	82.0	NaN	78
2	3	Student_3	100	88	71.0	99.0	-94
3	4	Student_4	72	99	NaN	84.0	86
4	5	Student_5	97	70	84.0	70.0	86
...	...	...	...	...	...	...	...
996	997	Student_997	88	68	84.0	66.0	98
997	998	Student_998	61	96	62.0	84.0	83
998	999	Student_999	72	76	90.0	72.0	90
999	1000	Student_1000	68	87	100.0	76.0	79
1000	1	Student_1	100	62	73.0	92.0	96

1001 rows × 7 columns

In [18]:
```
df.interpolate(method='linear',limit_direction = 'forward')
```

Out[18]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	1	Student_1	100	62	73.0	92.0	96
1	2	Student_2	72	97	82.0	95.5	78
2	3	Student_3	100	88	71.0	99.0	-94
3	4	Student_4	72	99	77.5	84.0	86
4	5	Student_5	97	70	84.0	70.0	86
...	...	...	...	...	...	...	...
996	997	Student_997	88	68	84.0	66.0	98
997	998	Student_998	61	96	62.0	84.0	83
998	999	Student_999	72	76	90.0	72.0	90
999	1000	Student_1000	68	87	100.0	76.0	79

1000	1	Student_1	100	62	73.0	92.0	96

1001 rows × 7 columns

In [19]:
```
df
```

Out[19]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	1	Student_1	100	62	73.0	92.0	96
1	2	Student_2	72	97	82.0	NaN	78
2	3	Student_3	100	88	71.0	99.0	-94
3	4	Student_4	72	99	NaN	84.0	86
4	5	Student_5	97	70	84.0	70.0	86
...	...	...	...	...	...	...	...
996	997	Student_997	88	68	84.0	66.0	98
997	998	Student_998	61	96	62.0	84.0	83
998	999	Student_999	72	76	90.0	72.0	90
999	1000	Student_1000	68	87	100.0	76.0	79
1000	1	Student_1	100	62	73.0	92.0	96

1001 rows × 7 columns

In [20]:
```
df.isnull().sum()
```

Out[20]:
```
Roll No       0
Name          0
Subject 1     0
Subject 2     0
Subject 3     1
Subject 4     1
Attendance    0
dtype: int64
```

In [21]:
```
df['Subject 3'].fillna(df['Subject 3'].mean(),inplace=True)
df['Subject 4'].fillna(df['Subject 4'].mean(),inplace=True)
df
```

Out[21]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	1	Student_1	100	62	73.00	92.000	96
1	2	Student_2	72	97	82.00	80.545	78
2	3	Student_3	100	88	71.00	99.000	-94
3	4	Student_4	72	99	79.89	84.000	86
4	5	Student_5	97	70	84.00	70.000	86
...	...	...	...	...	...	...	...
996	997	Student_997	88	68	84.00	66.000	98
997	998	Student_998	61	96	62.00	84.000	83
998	999	Student_999	72	76	90.00	72.000	90
999	1000	Student_1000	68	87	100.00	76.000	79
1000	1	Student_1	100	62	73.00	92.000	96

1001 rows × 7 columns

In [22]:
```
df.isnull().sum()
```

Out[22]:
```
Roll No       0
Name          0
Subject 1     0
Subject 2     0
Subject 3     0
Subject 4     0
Attendance    0
dtype: int64
```

replace()

In [23]:
```
df.replace(to_replace=np.nan,value=df['Subject 3'].mean(),inplace=True)
df
```

Out[23]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	1	Student_1	100	62	73.00	92.000	96
1	2	Student_2	72	97	82.00	80.545	78
2	3	Student_3	100	88	71.00	99.000	-94
3	4	Student_4	72	99	79.89	84.000	86
4	5	Student_5	97	70	84.00	70.000	86
...	...	...	...	...	...	...	...
996	997	Student_997	88	68	84.00	66.000	98
997	998	Student_998	61	96	62.00	84.000	83
998	999	Student_999	72	76	90.00	72.000	90
999	1000	Student_1000	68	87	100.00	76.000	79
1000	1	Student_1	100	62	73.00	92.000	96

1001 rows × 7 columns

Drop Missing Values

In [24]:
```
df = pd.read_csv("A1.csv")
df
```

Out[24]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	1	Student_1	100	62	73.0	92.0	96
1	2	Student_2	72	97	82.0	NaN	78
2	3	Student_3	100	88	71.0	99.0	-94
3	4	Student_4	72	99	NaN	84.0	86
4	5	Student_5	97	70	84.0	70.0	86
...	...	...	...	...	...	...	...
996	997	Student_997	88	68	84.0	66.0	98
997	998	Student_998	61	96	62.0	84.0	83
998	999	Student_999	72	76	90.0	72.0	90

999	1000	Student_1000	68	87	100.0	76.0	79
1000	1	Student_1	100	62	73.0	92.0	96

1001 rows × 7 columns

In [25]:
```
df.dropna()
```

Out[25]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	1	Student_1	100	62	73.0	92.0	96
2	3	Student_3	100	88	71.0	99.0	-94
4	5	Student_5	97	70	84.0	70.0	86
5	6	Student_6	98	76	89.0	92.0	82
6	7	Student_7	61	64	97.0	98.0	83
...	...	...	...	...	...	...	...
996	997	Student_997	88	68	84.0	66.0	98
997	998	Student_998	61	96	62.0	84.0	83
998	999	Student_999	72	76	90.0	72.0	90
999	1000	Student_1000	68	87	100.0	76.0	79
1000	1	Student_1	100	62	73.0	92.0	96

999 rows × 7 columns

In [26]:
```
df.shape
```

Out[26]:
```
(1001, 7)
```

In [27]:
```
## in a row if all features have null value then drop
df.dropna(how='all',inplace=True)
df
```

Out[27]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	1	Student_1	100	62	73.0	92.0	96
1	2	Student_2	72	97	82.0	NaN	78
2	3	Student_3	100	88	71.0	99.0	-94
3	4	Student_4	72	99	NaN	84.0	86
4	5	Student_5	97	70	84.0	70.0	86
...	...	...	...	...	...	...	...
996	997	Student_997	88	68	84.0	66.0	98
997	998	Student_998	61	96	62.0	84.0	83
998	999	Student_999	72	76	90.0	72.0	90
999	1000	Student_1000	68	87	100.0	76.0	79
1000	1	Student_1	100	62	73.0	92.0	96

1001 rows × 7 columns

In [28]:
```
## any value null drop
```

```
df.dropna(how='any',inplace=True)
df
```

Out[28]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	1	Student_1	100	62	73.0	92.0	96
2	3	Student_3	100	88	71.0	99.0	-94
4	5	Student_5	97	70	84.0	70.0	86
5	6	Student_6	98	76	89.0	92.0	82
6	7	Student_7	61	64	97.0	98.0	83
...	...	...	...	...	...	...	...
996	997	Student_997	88	68	84.0	66.0	98
997	998	Student_998	61	96	62.0	84.0	83
998	999	Student_999	72	76	90.0	72.0	90
999	1000	Student_1000	68	87	100.0	76.0	79
1000	1	Student_1	100	62	73.0	92.0	96

999 rows × 7 columns

In [29]:
```
## drop column
df.dropna(axis=1)
```

Out[29]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	1	Student_1	100	62	73.0	92.0	96
2	3	Student_3	100	88	71.0	99.0	-94
4	5	Student_5	97	70	84.0	70.0	86
5	6	Student_6	98	76	89.0	92.0	82
6	7	Student_7	61	64	97.0	98.0	83
...	...	...	...	...	...	...	...
996	997	Student_997	88	68	84.0	66.0	98
997	998	Student_998	61	96	62.0	84.0	83
998	999	Student_999	72	76	90.0	72.0	90
999	1000	Student_1000	68	87	100.0	76.0	79
1000	1	Student_1	100	62	73.0	92.0	96

999 rows × 7 columns

Check for negative Values

In [30]:
```
df[df[['Subject 1','Subject 2','Subject 3','Subject 4','Attendance']]<0]
```

Out[30]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	NaN	NaN	NaN	NaN	NaN	NaN	-94.0
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN
5	NaN	NaN	NaN	NaN	NaN	NaN	NaN

6	NaN	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...
996	NaN	NaN	NaN	NaN	NaN	NaN	NaN
997	NaN	NaN	NaN	NaN	NaN	NaN	NaN
998	NaN	NaN	NaN	NaN	NaN	NaN	NaN
999	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1000	NaN	NaN	NaN	NaN	NaN	NaN	NaN

999 rows × 7 columns

In [31]:
```
df[['Subject 1','Subject 2','Subject 3','Subject 4','Attendance']] = df[['Subject 1','Su
```

In [32]:
```
df
```

Out[32]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	1	Student_1	100	62	73.0	92.0	96
2	3	Student_3	100	88	71.0	99.0	0
4	5	Student_5	97	70	84.0	70.0	86
5	6	Student_6	98	76	89.0	92.0	82
6	7	Student_7	61	64	97.0	98.0	83
...	...	...	...	...	...	...	...
996	997	Student_997	88	68	84.0	66.0	98
997	998	Student_998	61	96	62.0	84.0	83
998	999	Student_999	72	76	90.0	72.0	90
999	1000	Student_1000	68	87	100.0	76.0	79
1000	1	Student_1	100	62	73.0	92.0	96

999 rows × 7 columns

Handling Inconsistencies - Duplicate Data

In [33]:
```
df.drop_duplicates(inplace=True)
df
```

Out[33]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	1	Student_1	100	62	73.0	92.0	96
2	3	Student_3	100	88	71.0	99.0	0
4	5	Student_5	97	70	84.0	70.0	86
5	6	Student_6	98	76	89.0	92.0	82
6	7	Student_7	61	64	97.0	98.0	83
...	...	...	...	...	...	...	...
995	996	Student_996	74	89	85.0	71.0	87
996	997	Student_997	88	68	84.0	66.0	98
997	998	Student_998	61	96	62.0	84.0	83

998	999	Student_999	72	76	90.0	72.0	90
999	1000	Student_1000	68	87	100.0	76.0	79

998 rows × 7 columns

Handling Outliers

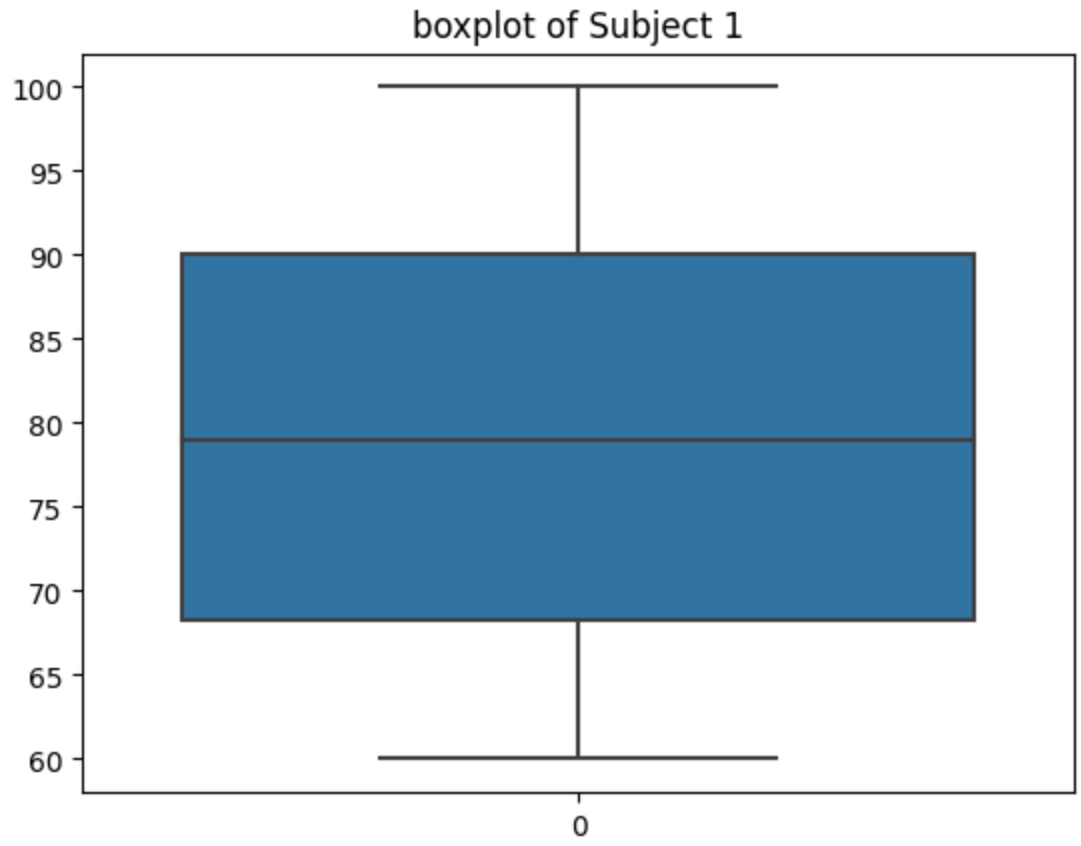
Visualization

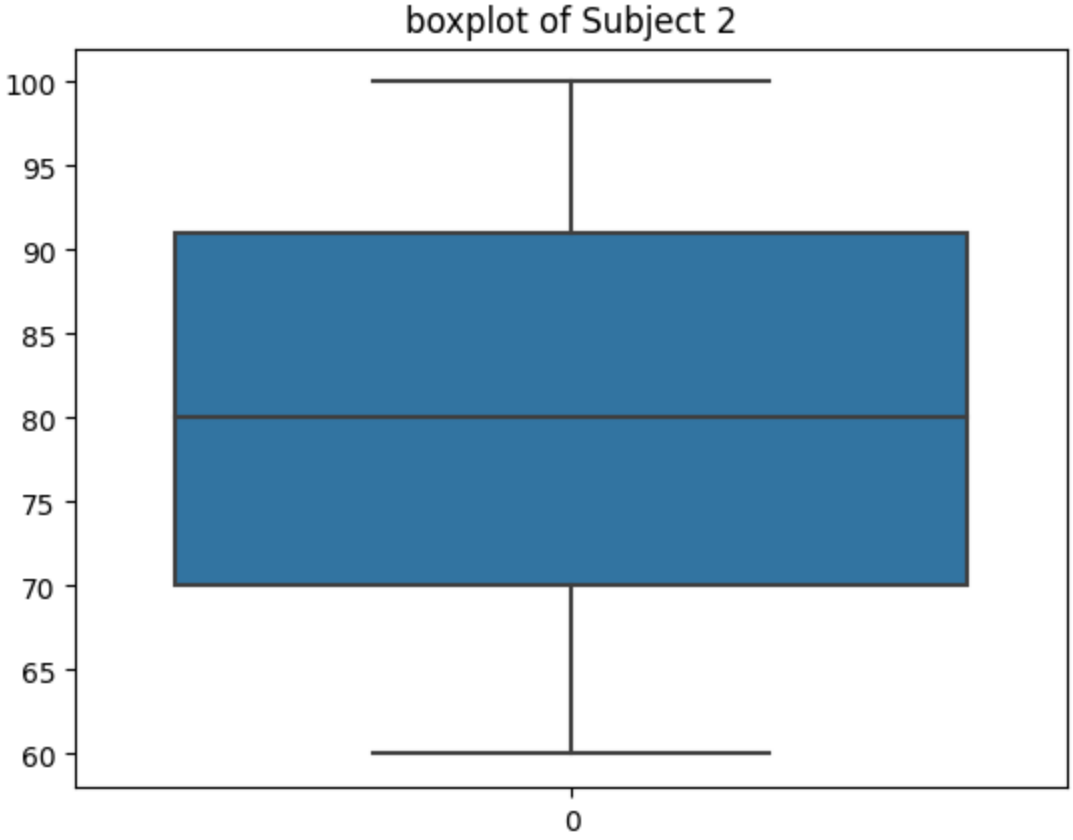
In [34]:
```
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

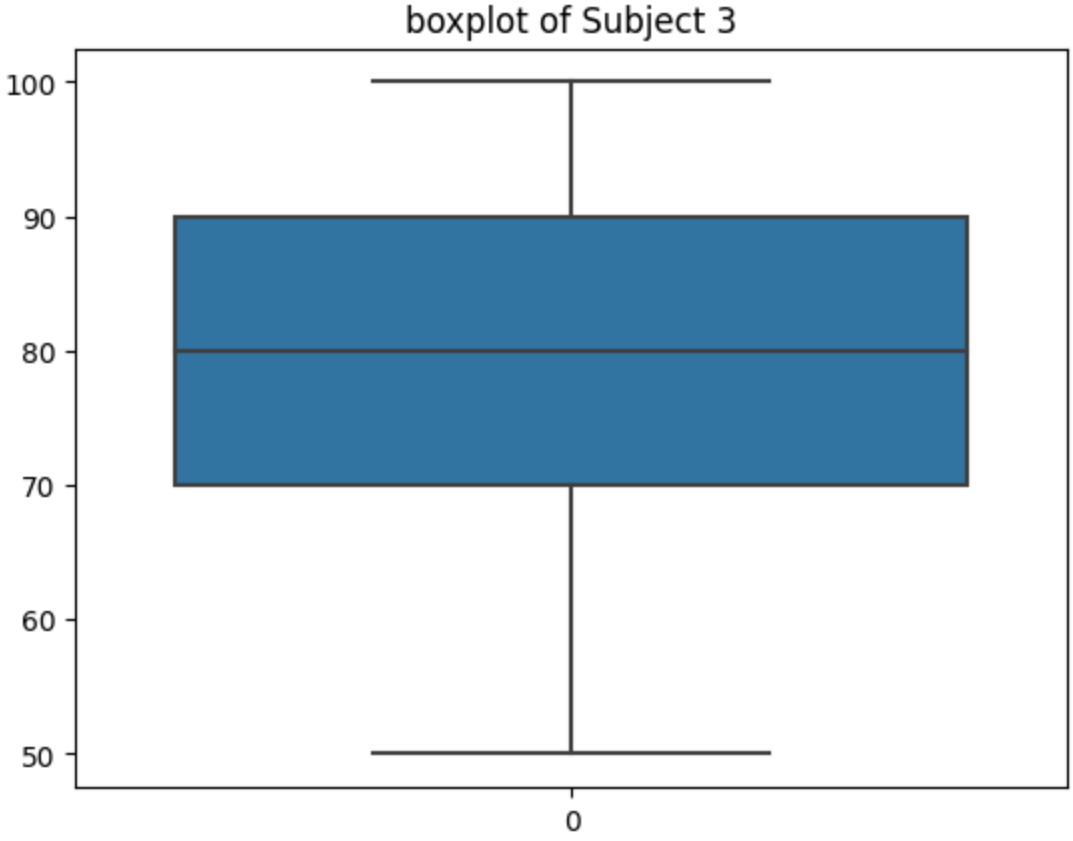
In [35]:
```
nc = ['Subject 1','Subject 2','Subject 3','Subject 4','Attendance']
nc
```

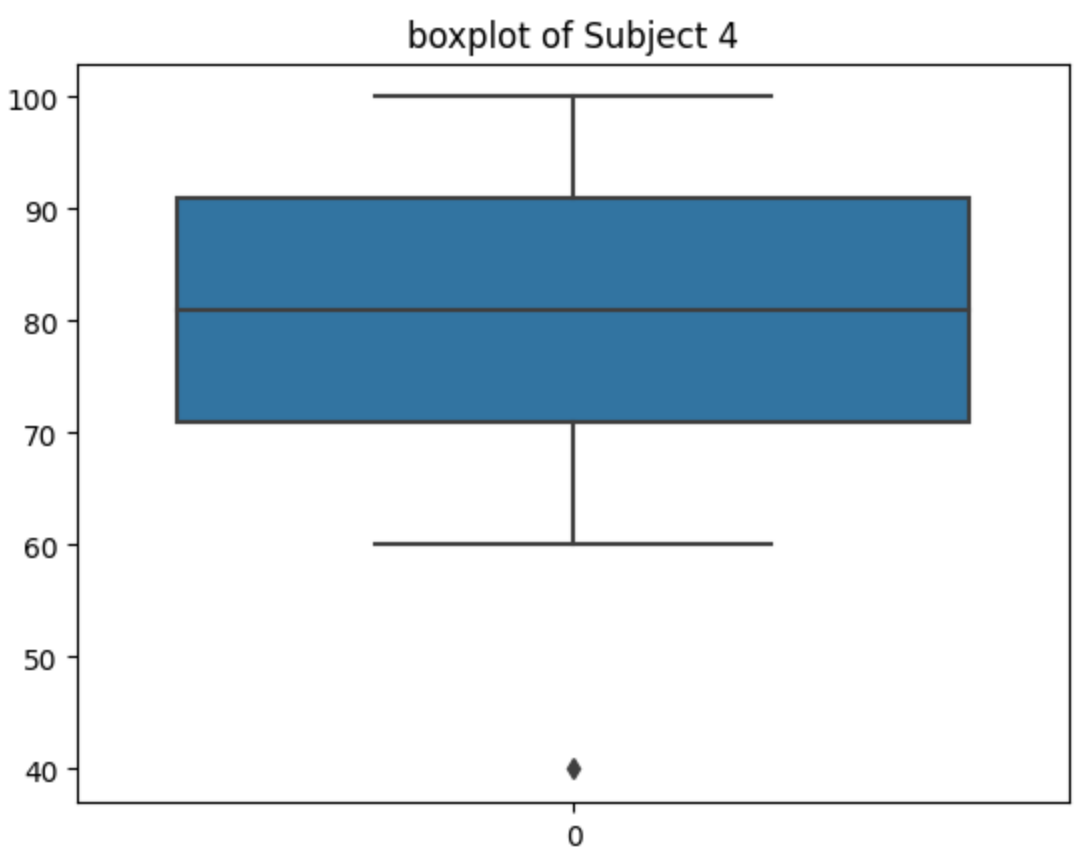
Out[35]:
```
['Subject 1', 'Subject 2', 'Subject 3', 'Subject 4', 'Attendance']
```

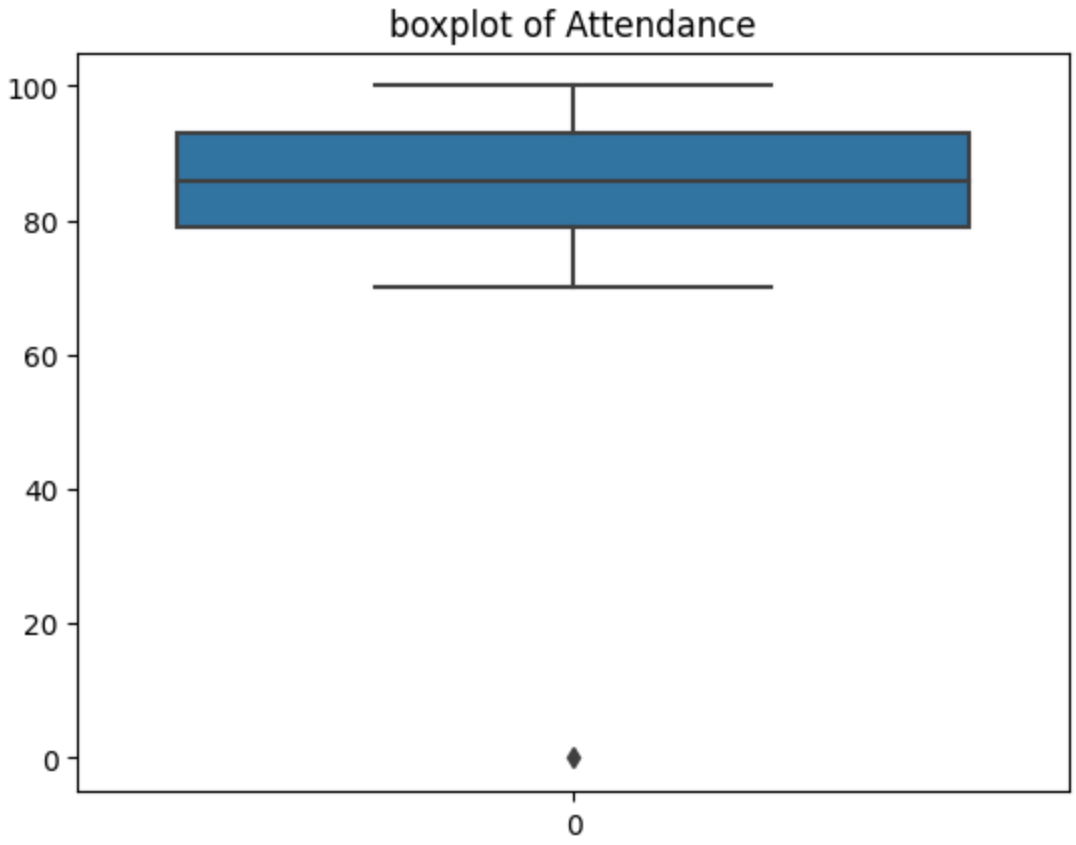
In [36]:
```
for col in nc:
    sns.boxplot(df[col])
    plt.title(f'boxplot of {col}')
    plt.show()
```



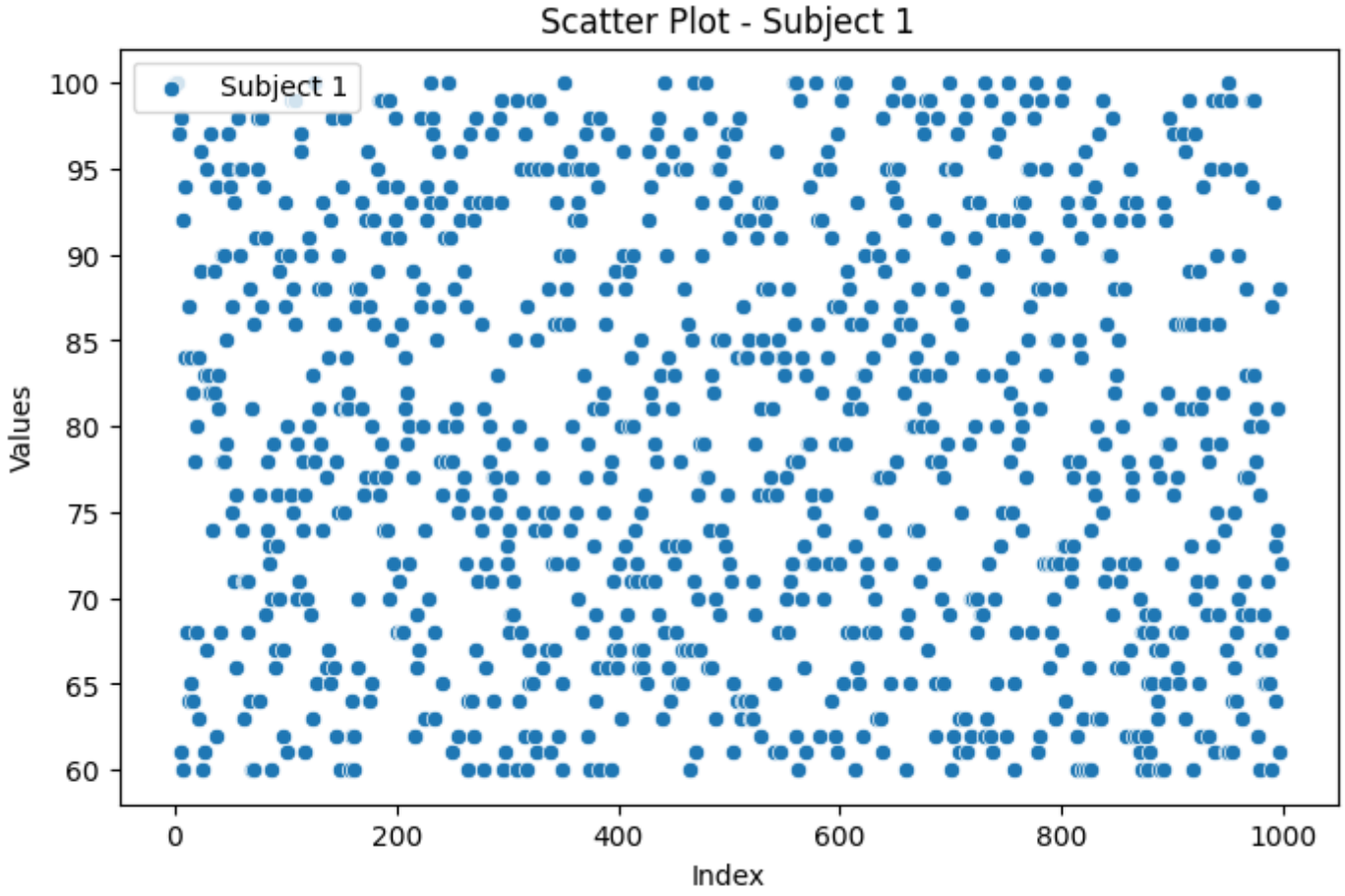

boxplot of Subject 2
100
95
90
85
80
75
70
65
60
0

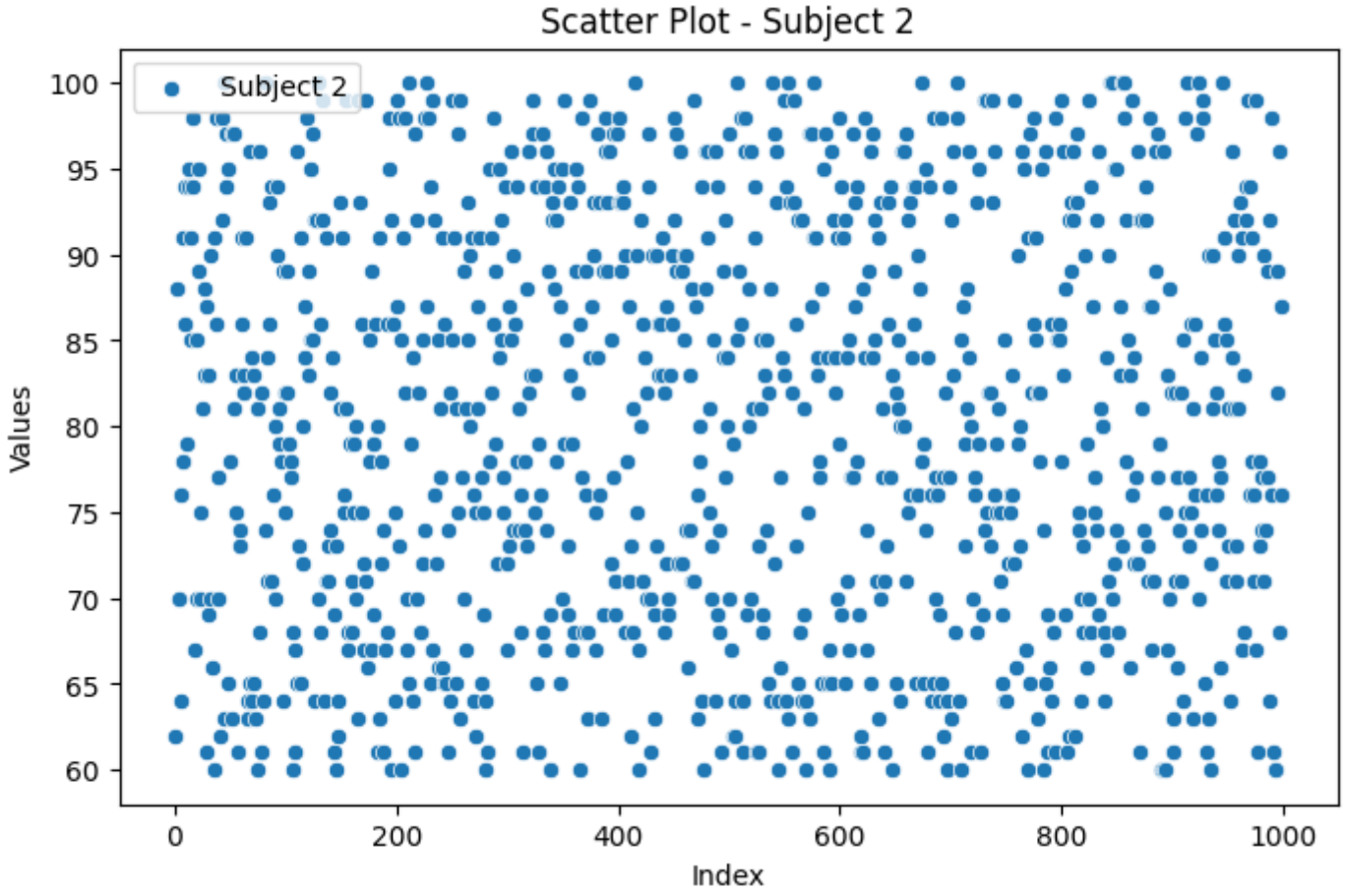

boxplot of Subject 3
100
90
80
70
60
50
0

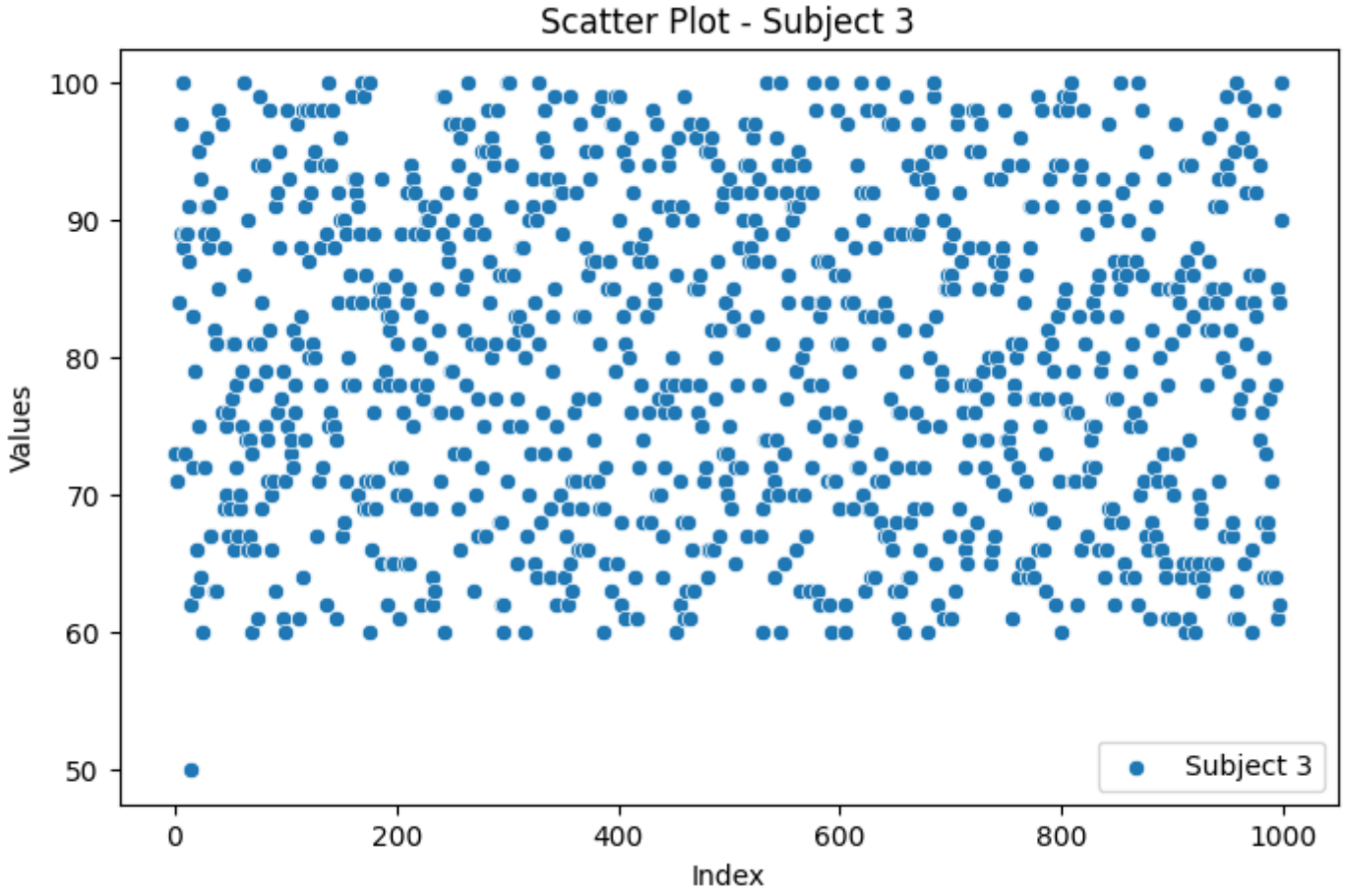


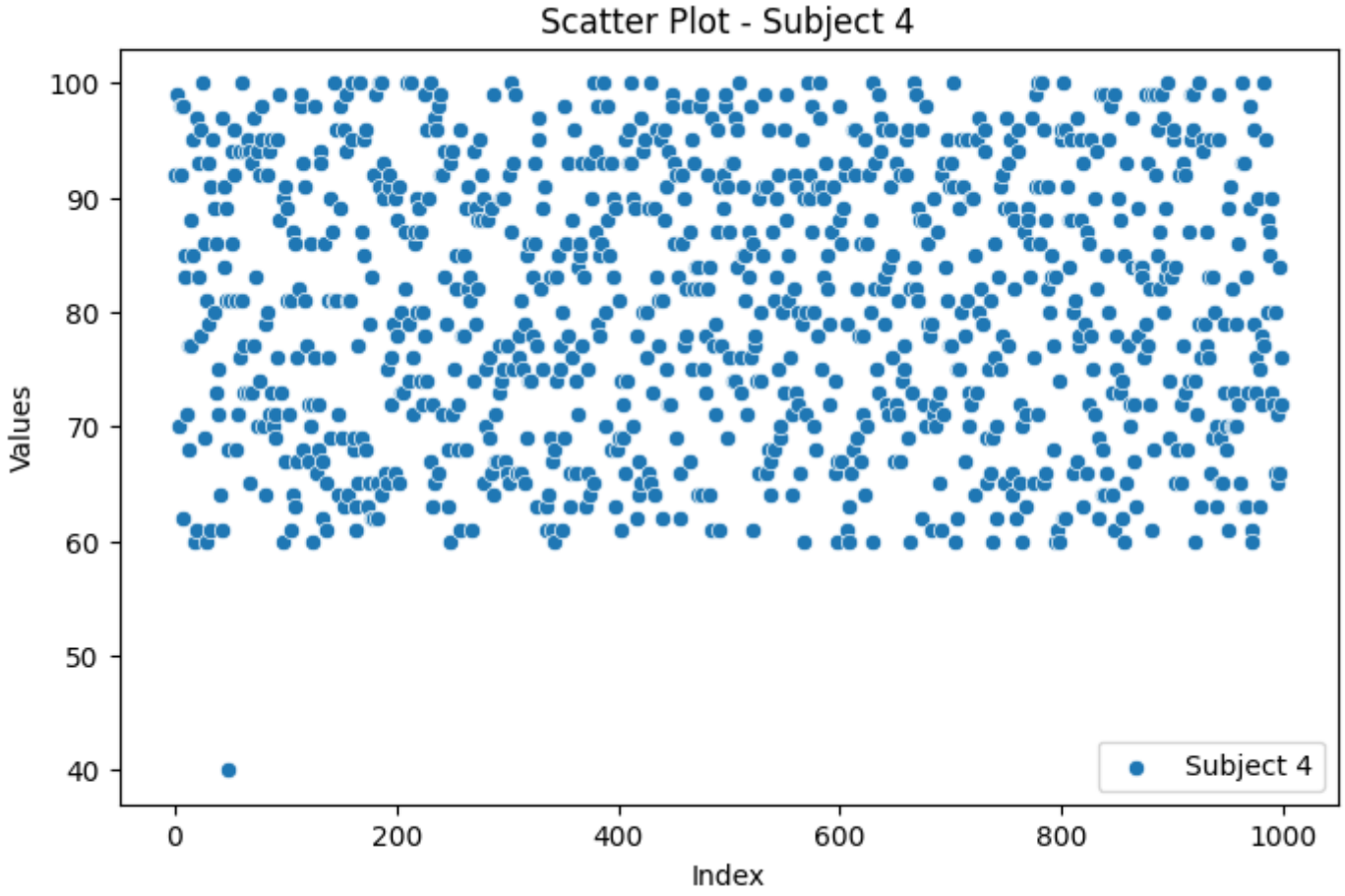


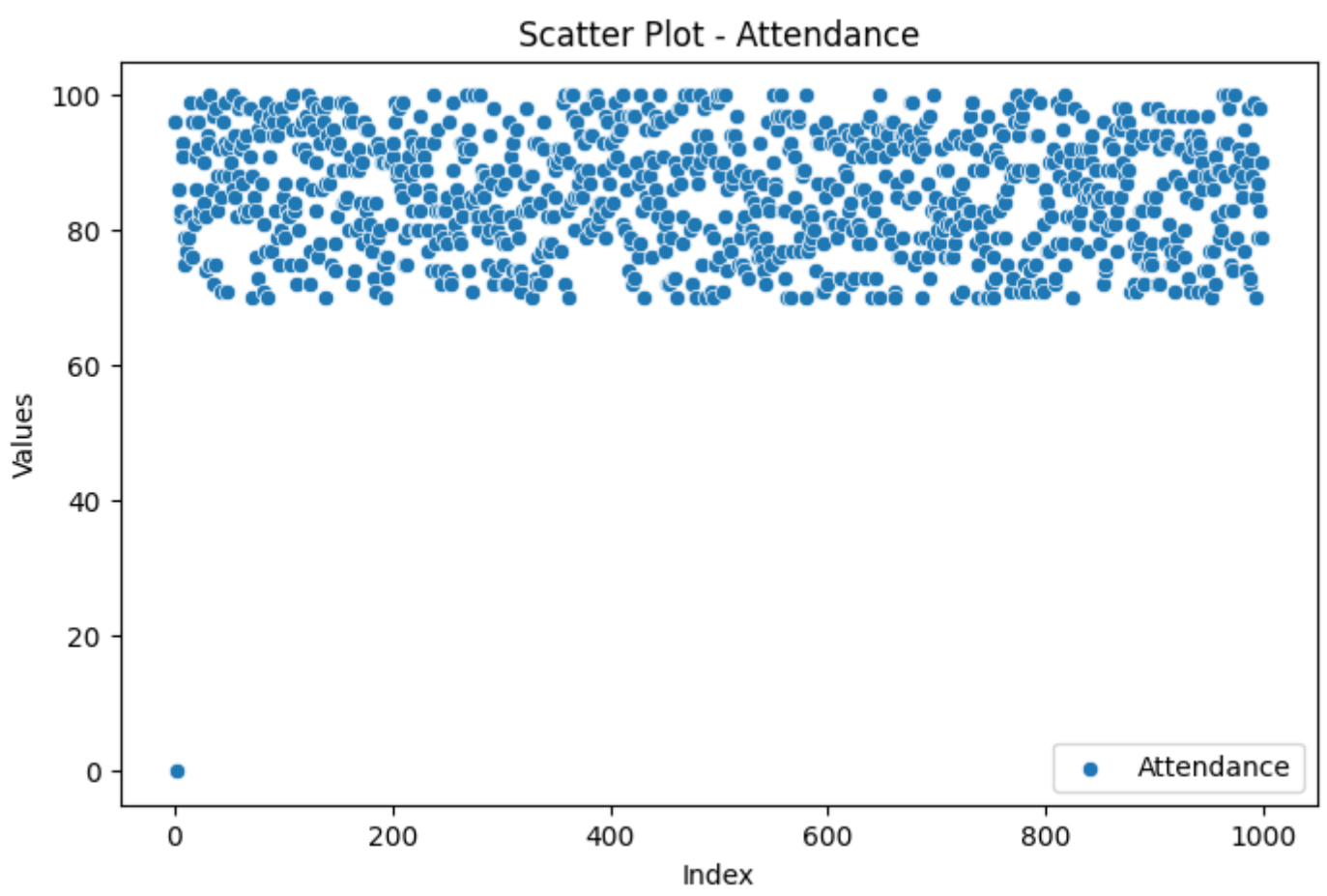
In [37]:
```
for column in nc:
    plt.figure(figsize=(8, 5))
    sns.scatterplot(data=df, x=df.index, y=column, label=column)

    plt.title(f'Scatter Plot - {column}')
    plt.xlabel('Index')
    plt.ylabel('Values')
    plt.legend()
    plt.show()
```


Scatter Plot - Subject 1
Subject 1
Values
Index
100
95
90
85
80
75
70
65
60
0
200
400
600
800
1000


Scatter Plot - Subject 2
Subject 2
Values
Index
100
95
90
85
80
75
70
65
60
0
200
400
600
800
1000


Scatter Plot - Subject 3
Values
Index
Subject 3
100
90
80
70
60
50
0
200
400
600
800
1000


Scatter Plot - Subject 4
Values
Index
Subject 4
100
90
80
70
60
50
40
0
200
400
600
800
1000



In [ ]:

Handling Outliers

In [38]:
```
for col in nc:
    Q1 = df[col].quantile(0.25)
    Q3 = df[col].quantile(0.75)
    IQR = Q3-Q1
    lb = Q1 - 1.5*IQR
    ub = Q3 + 1.5*IQR
    df[col] = np.where((df[col]<lb) | (df[col]>ub),df[col].median(),df[col])

df
```

Out[38]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance
0	1	Student_1	100.0	62.0	73.0	92.0	96.0
2	3	Student_3	100.0	88.0	71.0	99.0	86.0
4	5	Student_5	97.0	70.0	84.0	70.0	86.0
5	6	Student_6	98.0	76.0	89.0	92.0	82.0
6	7	Student_7	61.0	64.0	97.0	98.0	83.0
...	...	...	...	...	...	...	...
995	996	Student_996	74.0	89.0	85.0	71.0	87.0
996	997	Student_997	88.0	68.0	84.0	66.0	98.0
997	998	Student_998	61.0	96.0	62.0	84.0	83.0
998	999	Student_999	72.0	76.0	90.0	72.0	90.0
999	1000	Student_1000	68.0	87.0	100.0	76.0	79.0

998 rows × 7 columns

In [39]:
```
dfz = pd.read_csv('A1.csv')
```

In [40]:
```
#z-score manually
outliers = []
def detect_outliers_zscore(data):
    thres = 3
    median_value = data.median()
    mean = np.mean(data)
    std = np.std(data)
    # print(mean, std)
    for i in data:
        z_score = (i-mean)/std
        if (np.abs(z_score) > thres):
            i = median_value
    return data

for feat in nc:
    detect_outliers_zscore(dfz[feat])
```

Skewness

In [41]:
```
from scipy.stats import skew

for col in nc:
    skewness = skew(df[col])
    print(f"Skewness of {col}: {skewness}")
```

```
Skewness of Subject 1: 0.08854305280754458
Skewness of Subject 2: -0.009707975611008859
Skewness of Subject 3: 0.004967632869006547
Skewness of Subject 4: -0.035664011006138696
Skewness of Attendance: -0.09253182297757746
```

Data Transformation

In [42]:
```
df['log_attendence'] = np.log1p(df['Attendance'])
df
```

Out[42]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance	log_attendence
0	1	Student_1	100.0	62.0	73.0	92.0	96.0	4.574711
2	3	Student_3	100.0	88.0	71.0	99.0	86.0	4.465908
4	5	Student_5	97.0	70.0	84.0	70.0	86.0	4.465908
5	6	Student_6	98.0	76.0	89.0	92.0	82.0	4.418841
6	7	Student_7	61.0	64.0	97.0	98.0	83.0	4.430817
...	...	...	...	...	...	...	...	...
995	996	Student_996	74.0	89.0	85.0	71.0	87.0	4.477337
996	997	Student_997	88.0	68.0	84.0	66.0	98.0	4.595120
997	998	Student_998	61.0	96.0	62.0	84.0	83.0	4.430817
998	999	Student_999	72.0	76.0	90.0	72.0	90.0	4.510860
999	1000	Student_1000	68.0	87.0	100.0	76.0	79.0	4.382027

998 rows × 8 columns

In [43]:
```
df['sqrt_attendence'] = np.sqrt(df['Attendance'])
df
```

Out[43]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance	log_attendence	sqrt_attendence
0	1	Student_1	100.0	62.0	73.0	92.0	96.0	4.574711	9.797959
2	3	Student_3	100.0	88.0	71.0	99.0	86.0	4.465908	9.273618
4	5	Student_5	97.0	70.0	84.0	70.0	86.0	4.465908	9.273618
5	6	Student_6	98.0	76.0	89.0	92.0	82.0	4.418841	9.055385
6	7	Student_7	61.0	64.0	97.0	98.0	83.0	4.430817	9.110434
...	...	...	...	...	...	...	...	...	...
995	996	Student_996	74.0	89.0	85.0	71.0	87.0	4.477337	9.327379
996	997	Student_997	88.0	68.0	84.0	66.0	98.0	4.595120	9.899495
997	998	Student_998	61.0	96.0	62.0	84.0	83.0	4.430817	9.110434
998	999	Student_999	72.0	76.0	90.0	72.0	90.0	4.510860	9.486833
999	1000	Student_1000	68.0	87.0	100.0	76.0	79.0	4.382027	8.888194

998 rows × 9 columns

In [44]:
```
df['cbrt_Attendance'] = np.cbrt(df['Attendance'])
df
```

Out[44]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance	log_attendence	sqrt_attendence	cbrt
0	1	Student_1	100.0	62.0	73.0	92.0	96.0	4.574711	9.797959	
2	3	Student_3	100.0	88.0	71.0	99.0	86.0	4.465908	9.273618	
4	5	Student_5	97.0	70.0	84.0	70.0	86.0	4.465908	9.273618	
5	6	Student_6	98.0	76.0	89.0	92.0	82.0	4.418841	9.055385	
6	7	Student_7	61.0	64.0	97.0	98.0	83.0	4.430817	9.110434	
...	...	...	...	...	...	...	...	...	...	
995	996	Student_996	74.0	89.0	85.0	71.0	87.0	4.477337	9.327379	
996	997	Student_997	88.0	68.0	84.0	66.0	98.0	4.595120	9.899495	
997	998	Student_998	61.0	96.0	62.0	84.0	83.0	4.430817	9.110434	
998	999	Student_999	72.0	76.0	90.0	72.0	90.0	4.510860	9.486833	
999	1000	Student_1000	68.0	87.0	100.0	76.0	79.0	4.382027	8.888194	

998 rows × 10 columns

Insert New Column

In [45]:
```
data = {'Value': np.random.randn(df.shape[0])}  # Example data, you can replace it with
df_div = pd.DataFrame(data)
categories = ['A', 'B', 'C', 'D']
df_div['Div'] = np.random.choice(categories, size=len(df))
```

In [46]:
```
merged_df = pd.concat([df, df_div], axis=1)
merged_df
```

Out[46]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance	log_attendence	sqrt_attendence	cl
0	1.0	Student_1	100.0	62.0	73.0	92.0	96.0	4.574711	9.797959	
2	3.0	Student_3	100.0	88.0	71.0	99.0	86.0	4.465908	9.273618	
4	5.0	Student_5	97.0	70.0	84.0	70.0	86.0	4.465908	9.273618	
5	6.0	Student_6	98.0	76.0	89.0	92.0	82.0	4.418841	9.055385	
6	7.0	Student_7	61.0	64.0	97.0	98.0	83.0	4.430817	9.110434	
...	...	...	...	...	...	...	...	...	...	...
997	998.0	Student_998	61.0	96.0	62.0	84.0	83.0	4.430817	9.110434	
998	999.0	Student_999	72.0	76.0	90.0	72.0	90.0	4.510860	9.486833	
999	1000.0	Student_1000	68.0	87.0	100.0	76.0	79.0	4.382027	8.888194	
1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
3	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

1000 rows × 12 columns

In [47]:
```
merged_df = merged_df[:-2]
merged_df
```

Out[47]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance	log_attendence	sqrt_attendence	cl
0	1.0	Student_1	100.0	62.0	73.0	92.0	96.0	4.574711	9.797959	
2	3.0	Student_3	100.0	88.0	71.0	99.0	86.0	4.465908	9.273618	
4	5.0	Student_5	97.0	70.0	84.0	70.0	86.0	4.465908	9.273618	
5	6.0	Student_6	98.0	76.0	89.0	92.0	82.0	4.418841	9.055385	
6	7.0	Student_7	61.0	64.0	97.0	98.0	83.0	4.430817	9.110434	
...	...	...	...	...	...	...	...	...	...	...
995	996.0	Student_996	74.0	89.0	85.0	71.0	87.0	4.477337	9.327379	
996	997.0	Student_997	88.0	68.0	84.0	66.0	98.0	4.595120	9.899495	
997	998.0	Student_998	61.0	96.0	62.0	84.0	83.0	4.430817	9.110434	
998	999.0	Student_999	72.0	76.0	90.0	72.0	90.0	4.510860	9.486833	
999	1000.0	Student_1000	68.0	87.0	100.0	76.0	79.0	4.382027	8.888194	

998 rows × 12 columns

In [48]:
```
merged_df['Div'] = merged_df['Div'].fillna(merged_df['Div'].mode())

merged_df
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_25024\1273381511.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
  merged_df['Div'] = merged_df['Div'].fillna(merged_df['Div'].mode())
```

Out[48]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance	log_attendence	sqrt_attendence	cl
0	1.0	Student_1	100.0	62.0	73.0	92.0	96.0	4.574711	9.797959	
2	3.0	Student_3	100.0	88.0	71.0	99.0	86.0	4.465908	9.273618	
4	5.0	Student_5	97.0	70.0	84.0	70.0	86.0	4.465908	9.273618	
5	6.0	Student_6	98.0	76.0	89.0	92.0	82.0	4.418841	9.055385	
6	7.0	Student_7	61.0	64.0	97.0	98.0	83.0	4.430817	9.110434	
...	...	...	...	...	...	...	...	...	...	
995	996.0	Student_996	74.0	89.0	85.0	71.0	87.0	4.477337	9.327379	
996	997.0	Student_997	88.0	68.0	84.0	66.0	98.0	4.595120	9.899495	
997	998.0	Student_998	61.0	96.0	62.0	84.0	83.0	4.430817	9.110434	
998	999.0	Student_999	72.0	76.0	90.0	72.0	90.0	4.510860	9.486833	
999	1000.0	Student_1000	68.0	87.0	100.0	76.0	79.0	4.382027	8.888194	

998 rows × 12 columns

One Hot Encoding

In [49]:
```
#Performing Onehot Encoding
encoded_df = pd.get_dummies(merged_df['Div'])
```

In [50]:
```
merged_df = pd.concat([merged_df, encoded_df], axis=1)
```

In [51]:
```
merged_df
```

Out[51]:

	Roll No	Name	Subject 1	Subject 2	Subject 3	Subject 4	Attendance	log_attendence	sqrt_attendence	cl
0	1.0	Student_1	100.0	62.0	73.0	92.0	96.0	4.574711	9.797959	
2	3.0	Student_3	100.0	88.0	71.0	99.0	86.0	4.465908	9.273618	
4	5.0	Student_5	97.0	70.0	84.0	70.0	86.0	4.465908	9.273618	
5	6.0	Student_6	98.0	76.0	89.0	92.0	82.0	4.418841	9.055385	
6	7.0	Student_7	61.0	64.0	97.0	98.0	83.0	4.430817	9.110434	
...	...	...	...	...	...	...	...	...	...	
995	996.0	Student_996	74.0	89.0	85.0	71.0	87.0	4.477337	9.327379	
996	997.0	Student_997	88.0	68.0	84.0	66.0	98.0	4.595120	9.899495	
997	998.0	Student_998	61.0	96.0	62.0	84.0	83.0	4.430817	9.110434	
998	999.0	Student_999	72.0	76.0	90.0	72.0	90.0	4.510860	9.486833	
999	1000.0	Student_1000	68.0	87.0	100.0	76.0	79.0	4.382027	8.888194	

998 rows × 16 columns

In [ ]:
```
```